PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA
ESCUELA DE SISTEMAS

PROPUESTA DE GUÍA DE PROCESOS PARA DOCUMENTAR Y OPTIMIZAR EL DESARROLLO DE SISTEMAS REALIZADOS EN LOS LENGUAJES JAVA, RPG, COBOL-CICS Y LOTUS NOTES, UTILIZANDO LA METODOLOGÍA RUP, EN EL ÁREA DE SISTEMAS Y TELECOMUNICACIONES DE PETROCOMERCIAL

Previa a la obtención del Título de:

INGENIERO EN SISTEMAS E INFORMÁTICA

POR:

XAVIER RICARDO SALAZAR QUINTANA
EDUARDO ANDRÉS VELALCÁZAR ARMAS

QUITO, 16 de abril del 2012

ÍNDICE DE CONTENIDOS

LISTADO DE TABLAS

Marco Teórico

Situación Actual

RPG

COBOL Cics

LOTUS NOTES

JAVA

Fases, elementos y responsables del proceso de desarrollo de aplicaciones

Propuesta

Lenguajes RPG - COBOL Cics - LOTUS NOTES

Fases, elementos, responsables y roles en el proceso de desarrollo de aplicaciones

LISTADO DE GRÁFICOS

Marco Teórico

Situación Actual

PROPUESTA

DEDICATORIA

A mi familia y María, quienes respaldaron la consecución
de este logro constantemente desde los inicios,
A mis abuelos, para quienes tengo la mayor atención, consideración y estima,
A los amigos de siempre, los de antes, los que ya no están y los que vendrán.

Xavier R. Salazar Quintana

A Dios, por darme La fuerzas necesarias en los momentos
que más las necesité y bendecirme con la posibilidad de caminar a
su lado durante toda mi vida
A mi familia por su amor y apoyo incondicional,
A mis compañeros y amigos, por brindarme su amistad sincera,
A toda la comunidad universitaria por trabajar cada día
en la construcción de un mundo mejor.

Eduardo A. Velalcázar Armas

AGRADECIMIENTO

Nuestro más profundo agradecimiento a Dios por su compañía y bendiciones en todos los momentos de nuestras vidas.

A nuestros queridos padres por su fe incondicional.

A nuestro director, Oswaldo Espinosa, y a nuestros revisores, Suyana Arcos y Guido Ochoa, quienes con sus conocimientos, orientación, motivación y paciencia brindadas, fueron un pilar fundamental de nuestro aprendizaje durante nuestra carrera universitaria y durante el desarrollo y culminación de este trabajo.

A Martha, Elena, Marisol y Mario, parte fundamental de la estructura administrativa de la Facultad de Ingeniería, y todos quienes conforman la PUCE, por contribuir de una u otra manera en nuestro desarrollo profesional y personal.

A nuestros compañeros y amigos ya que de ellos hemos recibido motivación y apoyo incondicional.

Los Autores:
Eduardo Velalcázar
Xavier Salazar

RESUMEN

PETROCOMERCIAL, empresa estatal cuya misión es contribuir al desarrollo nacional mediante el abastecimiento eficiente y oportuno de los derivados del petróleo, se ha visto en la necesidad de ser una entidad competitiva ante la constante exigencia que se presenta en el ámbito empresarial y que ha dado lugar al surgimiento de nuevos requerimientos al momento de desarrollar sistemas y software de todo tipo. Esto a su vez ha provocado que las metodologías destinadas al perfeccionamiento de esta labor y estandarización de los procesos correspondientes, mejoren las propuestas de sus primeras versiones. Sin embargo, no todas las metodologías materializan sus expectativas.

La metodología R.U.P. es un conjunto de metodologías estándar, *adaptables al contexto y necesidades de cada organización,* que se utiliza para definir un flujo de trabajo organizado y bien documentado. Al ser un marco de referencia de procesos, *influenciado por patrones de Proceso / Análisis y bien documentado*, se convierte en una enorme base de conocimiento en Ingeniería del Software.

Esta metodología, caracterizada por ser iterativa e incremental (al contrario de muchas metodologías antecesoras, secuenciales en su gran mayoría) propone una alternativa de trabajo organizado y bien documentado para el desarrollo de Sistemas; además que define claramente lo que cada personaje involucrado en el proceso debe hacer, lo cual permite acceder a un seguimiento más práctico y real tanto de la evolución cuanto de los resultados parciales y totales del desarrollo.

Con lo expresado, el fin de este trabajo de investigación busca documentar los procesos inmersos en el desarrollo y mantenimiento de los sistemas y aplicaciones en la Unidad de Sistemas y Telecomunicaciones para toda la empresa.

CAPITULO I

GENERALIDADES

1.1- Introducción

PETROCOMERCIAL es la filial de PETROECUADOR, responsable del transporte, almacenamiento y comercialización de derivados de petróleo en el territorio nacional.[1]

La principal función de PETROCOMERCIAL es la de abastecer de combustibles al país, dentro de un mercado de libre competencia y administrar la infraestructura de almacenamiento y transporte de combustibles del Estado.

Tomando en consideración lo mencionado anteriormente, PETROCOMERCIAL brinda un servicio eficiente a la comunidad ecuatoriana ya que cuenta con una infraestructura interna moderna y posee tecnología de punta que es capaz de satisfacer las necesidades que se presentan diariamente dentro del contexto de la entidad, y de esta

---

[1] A mediados del 2011, PETROCOMERCIAL pasa a formar parte del grupo PETROECUADOR EP, que junta todas las filiales en una sola. Centralizando su administración. Sin embargo, los procesos que maneja cada filial se mantienen sin ninguna modificación más que el cambio de la figura administrativa.

manera entregar un mejor servicio a los clientes, en este caso es toda la población del territorio ecuatoriano.

PETROCOMERCIAL de acuerdo a las ventajas que presentan las nuevas metodologías de desarrollo de sistemas, ha tomado en consideración la necesidad de adoptar una metodología formal, la misma que se ha situado en el medio como una de las que genera mayor documentación y mejores índices de calidad, que mitiguen la pérdida de tiempo y economicen recursos económicos al momento de emprender mejoras en el futuro. RUP, al ser una metodología adaptable a la realidad de cada empresa, permite definir claramente procesos, responsables y documentación de manera óptima. Además que reduce las cotas de tiempo para la entrega de productos finales de software, ya que la programación de tiempo considera iteraciones.

Actualmente la metodología de desarrollo de aplicaciones y sistemas es informal y genera cierta documentación; sin embargo, ésta no es supervisada ni pasa por filtros de control que aseguren la consistencia que muestra el sistema, motivo por el cual se han tenido que realizar correcciones importantes en la marcha. Esto implica reducción de tiempo de análisis y desarrollos estructurados con errores potenciales inherentes.

Este proyecto de investigación ha sido requerido, solicitado a los autores y autorizado por el Jefe del Departamento de Sistemas y

Telecomunicaciones de PETROCOMERCIAL, bajo el memorando 2010-001-PCO-OfM1061, con fecha 10 de enero del 2010.

1.2- Justificación del Proyecto

El presente trabajo busca brindar una herramienta que mida el alcance, desarrollo y resultados, parciales y finales, de los proyectos de desarrollo con mayor eficacia para que las correcciones y generaciones de nuevos sistemas sean aplicados y desplegados, no en el menor tiempo posible, pero si con mayores niveles de calidad y reduciendo gastos o desperdicio de bienes tangibles (costos para la empresa) e intangibles (tiempo en la corrección de errores).

El avance tecnológico geométrico de la última década presenta herramientas, técnicas, metodologías, lenguajes, equipos,… en fin, un conjunto extenso de conocimiento desarrollado que permite mejorar el servicio e innovar los productos de software que diariamente se desarrollan. En base a este criterio, PETROCOMERCIAL y el área de Sistemas y Telecomunicaciones que poseen aplicaciones manipuladas por decenas y cientos de usuarios en todo el país, quieren aplicar las técnica modernas que generen aplicaciones de calidad.

Durante los años de estudios en la universidad, se aprenden conceptos y técnicas que se aplican en áreas como la de ingeniería del software, pero en nuestra vida práctica en desarrollo de software, es cuando realmente se entiende cómo se deben aplicar estos conceptos efectivamente; así también, se comprende la existencia de otras técnicas

y/o métodos que nos ayudan a minimizar los problemas de desarrollo de sistemas.

El entender el proceso de desarrollo y las herramientas que manejamos, nos con lleva a aplicar no solo los conceptos aprendidos en la universidad, sino también los aprendidos en el campo laboral. Esta es una razón por la cual se va a realizar esta investigación con la finalidad de brindar un aporte a la comunidad informática en el campo de la ingeniería de requerimientos con el fin de minimizar la cantidad de proyectos de software, que no llegan a cumplir en su totalidad con sus objetivos propuestos.

1.3- Objetivos de la Investigación

1.3.1- Objetivos General

Desarrollar y documentar una guía de procesos, como propuesta, para el despliegue de nuevos sistemas o aplicaciones, y para el mantenimiento de las ya existentes, basándose en la Metodología R.U.P.; reflejando la realidad del área de Sistemas y Telecomunicaciones de PETROCOMERCIAL.

1.3.2- Objetivos Específicos

- ✓ Realizar un estudio sobre la(s) metodología(s) que se utilizan actualmente en el área de Sistemas y Telecomunicaciones de PETROCOMERCIAL y analizar los resultados mediante un estudio comparativo con la Metodología R.U.P.

- ✓ Documentar los procesos, que no aún no han sido considerados, para entender cuál es el flujo de trabajo y actividades que se sigue y qué roles están involucrados en el desarrollo de software.

- ✓ Definir los puntos más trascendentes requeridos por la metodología planteada, como base para el emprendimiento de proyectos de desarrollo para la unidad de Sistemas y Telecomunicaciones de PETROCOMERCIAL.

- ✓ Analizar los resultados de la propuesta determinando, claramente, las ventajas y posibles contratiempos que se puedan presentar al adaptar la metodología al proceso de desarrollo de aplicaciones dentro de la unidad de Sistemas y Telecomunicaciones de PETROCOMERCIAL.

1.4- Alcance

La investigación incluye el levantamiento de datos del proceso de desarrollo de software para los sistemas desarrollados en Java, Cobol – CICS, RPG y Lotus Notes dentro de la Unidad de Sistemas y Telecomunicaciones de PETROCOMERCIAL; además se analizará los resultados para proceder a compararlos con la propuesta de la metodología RUP en el entorno de desarrollo de software con el fin de encontrar ventajas y falencias. Finalmente, se elaborará una guía de procesos para los sistemas que se desarrollan, que se convertirá en la propuesta o producto final del trabajo.

CAPITULO II

MARCO TEORICO

2.1. METODOLOGÍAS PARA DESARROLLO DE SOFTWARE

Una metodología de desarrollo de software es un conjunto de tareas que deben ser ejecutadas, en un orden específico preferiblemente, para que el proceso de desarrollo en sí sea eficaz.

En la línea del tiempo han surgido distintos tipos de metodologías que se han adaptado a la realidad tecnológica vigente. Sin embargo, todas las metodologías mantienen un objetivo en común que es el de optimizar todos los recursos para que puedan ser aprovechados de manera eficiente.

Cada persona es libre de escoger aquella metodología que se adapte de mejor manera a la necesidad o realidad de su compañía. Los criterios y parámetros para realizar esto son variados y no se podría hablar de estándares acertadamente. Sin embargo, todas las metodologías hacen énfasis en aspectos específicos lo que permite clasificarlas, de forma muy general, en secuenciales (estructuradas) e iterativas (orientadas a objetos); donde cada uno de estos aspectos puede ser interpretado como beneficio y debilidad, dependiendo de los criterios y condicionantes del proyecto que se quiere emprender.

Para proyectos que no deban demorar mucho es necesario adoptar una metodología práctica en la que el tiempo que se invierta en la realización de la solución informática, no sea mayor al tiempo empleado en generar la documentación sobre el proceso realizado. Por otro lado, tenemos aquellos proyectos más representativos y con un impacto bastante más importante, para los cuales se debe pensar en una estructura clara y bien documentada en la que cada módulo desarrollado converja en una totalidad o en un sistema y, así, evitar codificaciones aisladas que deban ser consideradas de manera separatista pese a que su naturaleza y principales características se asemejen con otros módulos, todo debido a una falta de estándares para la utilización de herramientas informáticas, programación deficiente de tiempos de entrega del producto, lenguajes de programación diversos e incompatibles, y una serie de condicionantes exógenos que intervienen en el proceso de desarrollo de software.

Entonces, la utilización de una metodología documentada viene a ser primordial para que el proyecto sea de calidad, y su presencia nos asegura la generación de directivas explícitas e implícitas para todo el proceso y, también, la de estándares que nos servirán para alcanzar y materializar las metas propuestas.

La contra partida, o el hecho de no utilizar una metodología de desarrollo, nos hace correr el riesgo de:

- Incumplir los plazos de entrega,

- Obtener productos de software con los que el usuario final muestre su inconformidad debido a que no representan ni pueden suplir la necesidad o requerimiento inicial,
- Invertir más recursos de los planificados (o no disponibles, muchas veces),
- Inyectar informalidad al proceso,

  entre otros.

Independientemente de las consecuencias, graves, medianas o menores, las metodologías de desarrollo de software brindan herramientas y procedimientos que socavan los inconvenientes para concebir un proyecto posterior a la finalización del que se esté desarrollando.

Así pues, las mejores prácticas de cada metodología se van adaptando a la realidad de la compañía según los lineamientos que dicta y a la continuidad con la que se realizan dichos lineamientos. Como producto de esto, se logra un refinamiento bidireccional en el que tanto la metodología como el proceso de desarrollo de software se enriquecen del empirismo producido y adoptan cambios.

En resumen, una metodología de desarrollo define:

- Estados, etapas o fases, incluyendo todos los criterios que se tomarán en cuenta para la transición de uno a otro
- Tareas, actividades.

- Roles, incluyendo el perfil necesario para cada rol y la interacción entre ellos
- Artefactos o elementos entregables (se puede incluir un estándar de cada elemento)
- Herramientas de control, seguimiento, medición y perfeccionamiento del proceso.
- Principios, criterios para tomar decisiones, estrategias para manejar distintos tipos de situaciones, herramientas de manejo de riesgos, etc.

No debe confundirse metodología con ciclo de vida.

2.1.1. METODOLOGÍAS ITERATIVAS

- **GENERALIDADES**

Las metodologías iterativas proponen una forma de trabajo en la que:

- el proyecto de desarrollo puede ser dividido en iteraciones. El producto entregable no es más que una versión funcional del mismo sistema
- todos los elementos del desarrollo, o fases, están sujetos a tener modificaciones en las iteraciones posteriores
- se busca minimizar el impacto de los errores y no su eliminación
- el equipo de desarrollo aprovecha el aprendizaje y se tiene una retro alimentación más efectiva
- no se tenga dependencia al código fuente y su desarrollador, sino que exista un notable incremento de la portabilidad y reusabilidad de los componentes del sistema

La metodología iterativa permite no sólo revisar las etapas anteriores sino que también complementarlas si hiciera falta. Además, divide grandes proyectos en pequeñas piezas para ser realizadas mediante un proceso iterativo de análisis, diseño, desarrollo y pruebas.

- **ITERACIONES**

Las iteraciones contempla todos los pasos necesarios que permiten enfocar un subconjunto de elementos del proyecto completo de tal forma que se pueda finalizar en detalle.

Una característica de los proyectos de desarrollo de software que incluyen una metodología iterativa es que una vez que se ha puesto en marcha la fase de desarrollo, se encuentran nuevos requerimientos o se tiene la necesidad de redefinir los existentes. Sin embargo, no se tiene que desechar el avance conseguido, ya que dichos requerimientos pueden ser incorporados en una siguiente iteración.

De esta forma conseguimos realizar pruebas objetivas de funcionalidad para el subsistema de forma independiente y, así, el riesgo se reduce significativamente y el alcance del paso final no va más allá de conseguir la integración de cada parte funcional.

- FASES DE UNA ITERACIÓN

Las fases de un proyecto que aplica una metodología iterativa son Iniciación, Elaboración, Construcción y Transición.

Fase de Iniciación

Durante esta fase se debe explicar de lo que se trata el proyecto y la mejor forma de llevarlo a cabo. Los objetivos son, entre otros:

- Entender lo que se va a construir
- Identificar los puntos claves del proyecto
- Definir, al menos, una posible solución
- Entender cuáles son los costes, tiempos, y los riesgos del proyecto
- Preparar el ambiente base o de soporte para el proyecto

Como principales actividades o indicadores mínimos a conseguir para cumplir con los objetivos del ciclo de vida, tenemos que realizar un:

- consenso con el cliente sobre el alcance del proyecto y las estimaciones
- consenso en la identificación de los requisitos claves del proyecto
- consenso en las prioridades, los riesgos y el proceso de desarrollo definidos, y calificarlos de apropiados
- consenso en la identificación de los riesgos iniciales

Al final de la fase de iniciación debemos tener una idea bastante acertada del alcance del proyecto. Los detalles no serán obtenidos probablemente pero la visión general estará definida lo suficientemente clara que podremos hacer la primera división del proyecto en partes encapsuladas que permitan su creación independiente y que, además, satisfaga un subconjunto del requerimiento del sistema total.

Fase de Elaboración

Antes de elaborar la solución se deberá definir las actividades que controlará nuestro producto de software, qué tecnologías se deberán utilizar y cómo se piensa construir el sistema. Los objetivos son, entre otros:

- Identificar y describir gran parte de los requisitos
- Diseñar, implementar, revisar y validar la arquitectura
- Eliminar los riesgos más importantes y actualizar la planificación
- Refinar el proceso y configurar las herramientas a usar

El mayor enfoque para esta fase debe estar centrado en la identificación de los riesgos con los que se va a enfrentar; esto implica un análisis de la potencialidad de cada riesgo en la totalidad de un posible problema para clasificar los riesgos de mayor impacto y ponerles especial atención.

La detección de riesgos debe preceder a la realización de las etapas de análisis y diseño. Para esto se deben realizar dos pasos importantes:

- el modelo de dominio del sistema, que representa, de manera general, las operaciones del negocio en un diagrama, y
- el diagrama de los casos de uso, que son las interacciones del usuario con el sistema

- **Modelo de dominio**

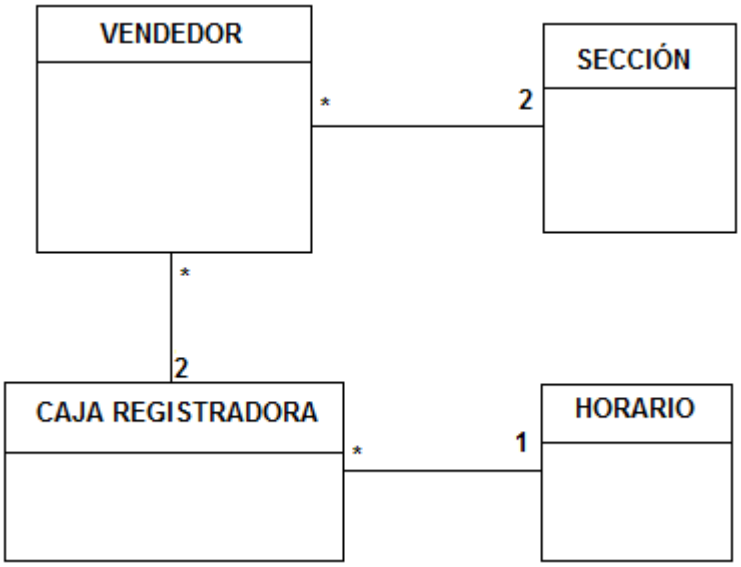


Gráfico 2.1. Modelo de dominio de una tienda de abarrotes
Autores: Xavier Salazar / Eduardo Velalcazar

El modelo de dominio describe el ambiente bajo el que funcionará el sistema y contiene poca cantidad de detalles como diagramas, conexiones, notas, comentarios, o los componentes que se consideren necesarios, que nos servirá para realizar un modelo más detallado y preciso.

En el Gráfico 2.1. se muestra una visión general de cómo funciona un negocio pequeño de venta de abarrotes. Si bien es cierto se omiten los detalles o campos de cada entidad, se muestra, en cambio, la relación que existe entre ellas, por lo que podemos ver que un número de vendedores se ocupan de las 4 secciones del local y de las 2 cajas registradoras, que funcionan en un horario específico. El éxito de este enfoque es que se conseguirá entender las dimensiones del sistema y el impacto de cada parte en del mismo. Luego procedemos con la diagramación de los casos de uso.

- **Casos de uso**

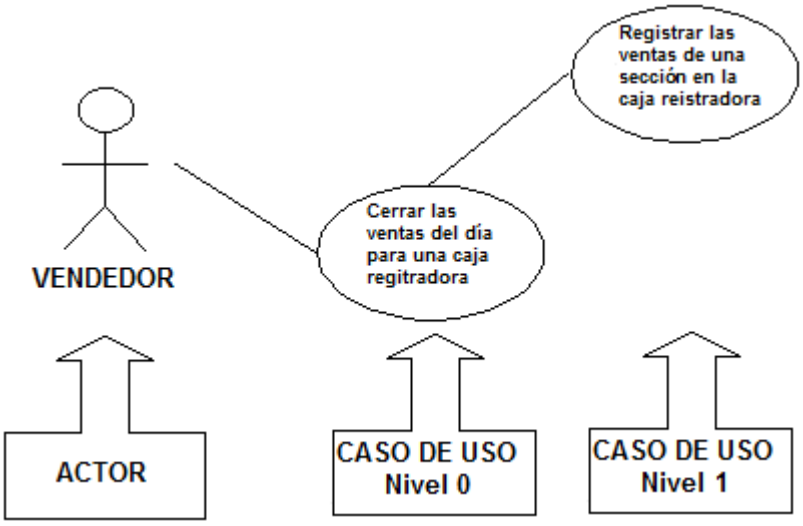


Gráfico 2.2. Diagrama de Casos de Uso de la tienda de abarrotes
Autores: Xavier Salazar / Eduardo Velalcazar

Los Casos de Uso describen las acciones que se realizan desde el punto de vista del usuario. Los diagramas pueden llegar a ser tan precisos y específicos como lo requiera el usuario. Para el ejemplo un caso de uso general es el de Cerrar las Ventas del Día y uno específico es el de Registrar las Ventas de una Sección. La descripción o el texto que nos sirva para describir el caso de uso debe ser lo suficientemente explícito para que los usuarios comprendan la idea y para que el equipo de desarrollo entienda el impacto del caso en la implementación y la funcionalidad del sistema.

El siguiente paso es realizar el modelo de diseño.

- **Modelo de Diseño**

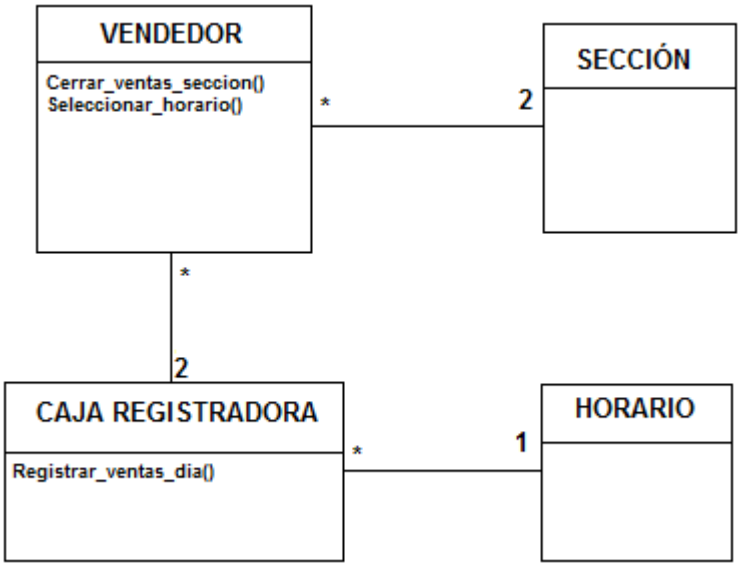


Gráfico 2.3. Modelo de Diseño de la tienda de abarrotes
Autores: Xavier Salazar / Eduardo Velalcazar

El modelo de diseño identifica, y concilia en un diagrama, a los objetos que el sistema contendrá y los casos de uso que se automatizarán pero sin invertir un nivel muy profundo de detalles ya que esto se hará en la etapa de desarrollo. Lo que si debe tener el modelo es una arquitectura reutilizable que permita para futuras extensiones del sistema.

La fase de elaboración de un proyecto se completa cuando se han identificado las tecnologías y todos los riesgos, y los planes para mitigarlos, se han creado los modelos del dominio, diagramas de casos de uso y modelo de diseño. Además, los desarrolladores deben desarrollar un cronograma para la creación de cada caso de uso o sub proyecto. Allí, la iteración juega un rol importante ya que durante la construcción vamos a crear cada caso de uso por separado y vamos a permitir que la experiencia obtenida en la creación de uno nos sirva para la creación del siguiente.

Como principales actividades o indicadores mínimos a conseguir para cumplir con los objetivos del ciclo de vida, tenemos que:

- tener un documento de requisitos y arquitectura estables,
- haber probado los prototipos para demostrar que los riesgos más importantes ya han sido mitigados,
- tener una proporción aceptable entre los recursos empleados frente a los estimados,
- tener una planificación, tratable, de las iteraciones para la siguiente fase,
- tener la aprobación del cliente para todo lo anterior

Fase de Construcción

En la fase de construcción se tratan los casos de uso mediante un análisis, diseño, codificación, pruebas y proceso de iteración que es donde se entrega un producto funcional terminado. Los objetivos principales, entre otros, son: minimizar los costes, obtener cierto grado de paralelismo en el desarrollo, y construir, iterativamente, una beta funcional del producto.

Es común encontrar nuevos casos de uso durante la construcción de un caso en específico, o la necesidad de redefinir los casos de uso realizados durante la fase de elaboración. Así pues, es posible realizar la planificación respectiva para su inclusión en una construcción posterior. Una vez que todos los casos de uso han sido construidos se realiza la integración entre cada uno de ellos dentro del sistema. Las revisiones a cada módulo, el nacimiento de nuevos sub proyectos, o la necesidad de cambios a los ya terminados, tienen la

posibilidad de re ingresar al proceso de construcción, el mismo que da el enfoque iterativo al proceso en general. Como resultado, se tiene una serie de iteraciones en cada subsistema.

Fase de Transición

La fase de transición centra sus esfuerzos en administrar los distintos aspectos que no fueron tratados durante la construcción, esto es: alguna integración que no haya sido considerada, la optimización del rendimiento, entre otros. Los principales objetivos, entre otros, son:

- Realizar una prueba del producto construido
- Formar los usuarios y encargados del mantenimiento
- Preparar la implementación
- Versionar el producto para definir el prototipo
- Realizar la prueba de aceptación del producto
- Obtener información para la mejora de futuros proyectos

La optimización al proceso debe darse luego de la evaluación del usuario y de la detección de las partes en las que el sistema procesa la información con algún tipo de retardo, que logra hacer el usuario. En caso contrario se invertirá mayor tiempo y esfuerzo en lugares equívocos. La etapa de transición es considerada, entonces, como el período de tiempo entre la libración de una versión beta y la versión final del proyecto. El enfoque de desarrollo iterativo nos permite facilitar el proceso de este nuevo caso de uso y luego re-entrar en la fase de transición.

2.1.1.1. Metodología R.U.P. (Rational Unified Process)

- **Conceptos relacionados**

Antes de hablar de la metodología RUP, es necesario tener claro los siguientes conceptos:

- Proceso: Define *quién* hace *qué*, *cuándo* lo hace y *cómo* alcanza un objetivo en específico [1]
- Método: Es el encuentro del proceso, la notación y la heurística
- Modelo: Es una abstracción que describe uno o más aspectos de un problema o de una posible solución anexando el problema en sí [2]
- Arquitectura: “La arquitectura de software de un programa o de un sistema computacional es la estructura o estructuras de un sistema, los cuales consta de elementos de software, propiedades visibles externamente de dichos elementos y las relaciones existentes entre ellos.” (Software Architecture in Practice, Second Edition/Len Bass, Paul Clements, Rick Kazman/ISBN 0-321-15495-9)

Principios de RUP

La metodología RUP se maneja bajo seis principios principales:

- Adaptar el proceso de desarrollo a las características propias del proyecto u organización
- Balancear las prioridades, como son los requerimientos contradictorios o diferentes, y la disputa de recursos limitados

1 Ivar Jacobson (Rational Software)
2 *Scott Ambler (AmbySoft)*

- Colaboración entre equipos, para que la comunicación sea fluida y la coordinación entre requerimientos, desarrollo, resultados, evaluaciones, planes, etc. resulte más sencilla
- Demostrar valor iterativamente mediante el análisis de la opinión de los inversores y de la estabilidad, y calidad, del producto; así como del refinamiento de la dirección del proyecto
- Elevar el nivel de abstracción con el uso de conceptos reutilizables (Patrón del software, Lenguajes 4GL, Esquemas, entre otros)
- Enfocarse en la calidad, pero no al final únicamente sino en todos los aspectos de la producción

Definiciones Generales

RUP fue desarrollado por Rational (IBM) y es definido como un conjunto de metodologías estándar adaptables al contexto y necesidad o necesidades de cada organización que se utiliza para sistemas orientados a objetos en los que se incluyan procesos de análisis, implementación y documentación. También, es un marco de referencia de procesos influenciado por patrones de Proceso / Análisis, y bien documentado, que es considerado una enorme base de conocimiento para la ingeniería del software.

Elementos principales de RUP

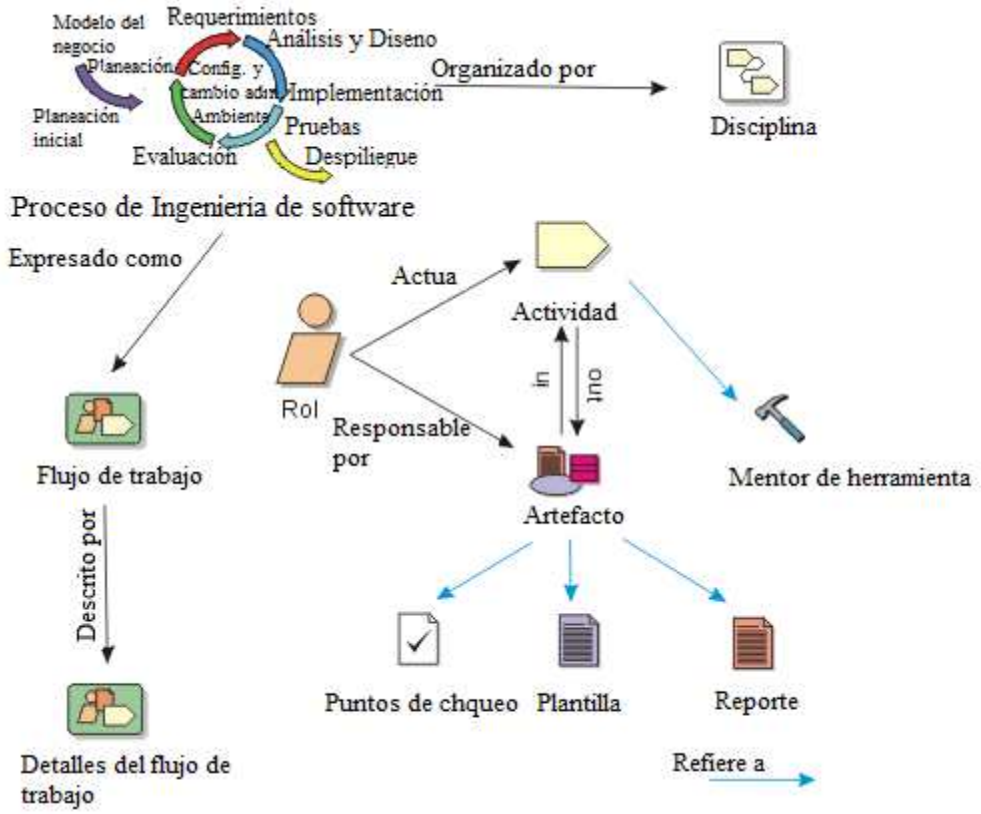


Gráfico 2.4. Elementos básicos de RUP
Editado de: Rational Software – Rational Unified Process

Éstos son unos ejemplos de los elementos que propone RUP para definir:

- Quién

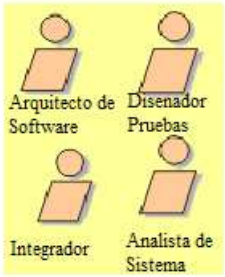


Gráfico 2.5. Ejemplo de algunos roles que propone RUP
Editado de: Rational Software – Rational Unified Process

- hace qué,:

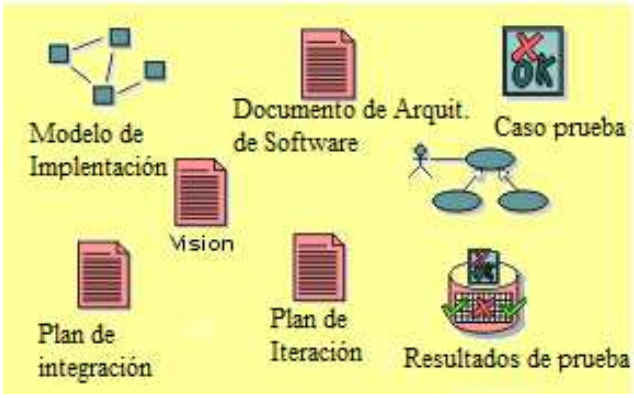


Gráfico 2.6. Ejemplo de algunos artefactos que propone RUP
Editado de: Rational Software – Rational Unified Process

- cómo lo hace:

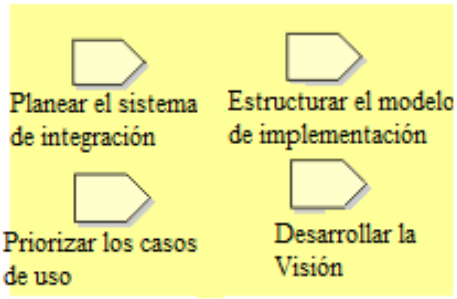


Gráfico 2.7. Ejemplo de algunas actividades que propone RUP
Editado de: Rational Software – Rational Unified Process

- y cuándo lo hace:

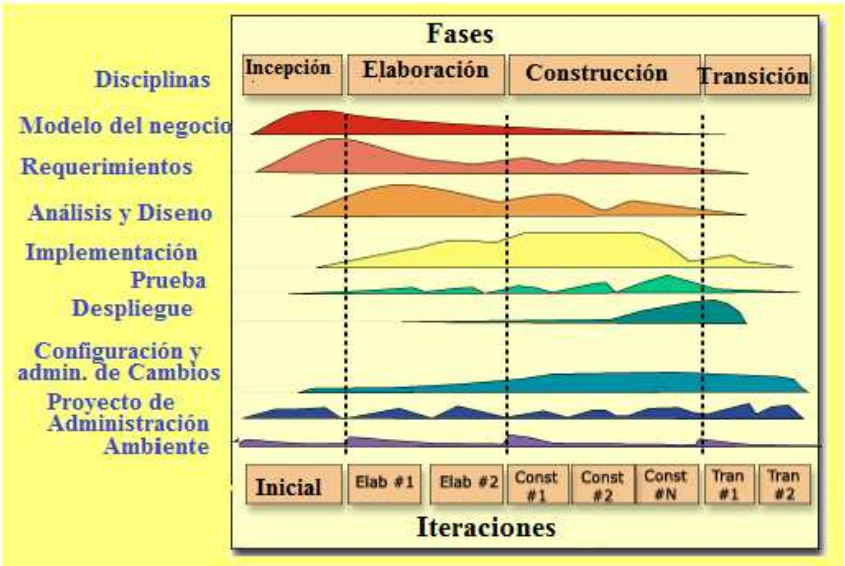


Gráfico 2.8. Estructura de evolución de RUP
Editado de: Rational Software – Rational Unified Process

RUP es un proceso de ingeniería de software que permite mejorar la productividad del equipo de trabajo y que, además, entrega las mejores prácticas del uso del software a todos sus miembros. Además, proporciona una guía específica en el Modelado de Negocios, adición de parámetros de calidad, generación de documentación, entre otros más.

Estructura de RUP

RUP se caracteriza por trabajar bajo dos dimensiones: estáticas y dinámicas.

1. **Estructura Estática**

Hace referencia a todos los elementos que forman parte de la metodología, como son los artefactos, los roles, disciplinas y flujos de trabajo, tratándolos a cada uno de manera individual y completa.

Artefactos

Los artefactos son considerados los requisitos previos y posteriores a la realización de las actividades, y vienen a ser resultados parciales o finales de la producción del software que son usados durante todo el proyecto. Un artefacto, entonces, puede ser: un documento, un modelo o un elemento de modelo. Los conjuntos de artefactos por cada disciplina son: *Business Modeling Set, Requirements Set, Analysis & Design Set, Implementation Set, Test Set, Deployment Set, Project Management Set, Configuration & Change Management Set,* y *Environment Set.*

Roles

Los roles son representaciones de los perfiles de usuario que van a estar encargados de administrar cada tarea o actividad. Todos los roles tienen una descripción exhaustiva de las cualidades que deben presentar y de las responsabilidades asociadas a ellos. Se pueden clasificar los roles en administradores, analistas, desarrolladores, profesionales para hacer pruebas y otros un poco más específicos. Este es el listado de los roles que presenta RUP por cada clasificación:

- Analyst: Business-Process Analyst, Business Designer, Business-Model Reviewer, Requirements Reviewer, System Analyst, Use-Case Specifier, User-Interface Designer
- Developer: Architect, Architecture Reviewer, Capsule Designer, Code Reviewer, Database Designer, Design Reviewer, Designer, Implementer, Integrator
- Testing professional: Test Designer, Tester
- Manager: Change Control Manager, Configuration Manager, Deployment Manager, Process Engineer, Project Manager, Project Reviewer
- Other: Course Developer, Graphic Artist, Stakeholder, System Administrator, Technical Writer, Tool Specialist

Disciplinas

Los objetivos de cada disciplina considerada en RUP son:

- Modelado del Negocio:
    - Entender la estructura y dinámica de la organización que albergará el sistema
    - Entender los problemas actuales en la organización e identificar posibles mejoras
    - Asegurar que los clientes, usuarios finales de la aplicación y los miembros del equipo de desarrollo tengan una misma visión acerca de la organización
    - Obtener requisitos necesarios para soportar la organización del cliente

- Requerimientos:
    - Establecer y mantener un acuerdo con los clientes y usuarios acerca de lo que el sistema debería hacer
    - Entender los requerimientos del sistema
    - Definir el alcance que va a tener el proyecto
    - Base para la estimación (iteraciones, coste y tiempo necesario para desarrollar el sistema)
- Análisis y Diseño:
    - Transformar los requisitos en modelos de diseño
    - Desarrollar una arquitectura robusta para el sistema
    - Adaptar el diseño al entorno de implementación

- Implementación:
    - Implementar los elementos de diseño
    - Test unitarios de los elementos implementados
    - Integración
- Pruebas:
    - Encontrar y documentar fallosen el software(UnitTest, IntegrationTest, SystemTest, AcceptanceTest) -> Calidad
    - Validar que el software tiene la funcionalidad diseñada
    - Validar que los requisitos han sido implementados de forma apropiada
- Despliegue:
    - Asegurar que el producto de software está listo para sus usuarios finales
- Gestión del proyecto:
    - Planificar, seleccionar el grupo de trabajo, ejecutar y supervisar el proyecto
    - Gestión de los riesgos
- Entorno:
    - Gestionar el entorno de desarrollo
    - Configurar y mejorar el proceso
    - Configurar herramientas
    - Brindar servicios técnicos para que se pueda soportar el proceso (Infraestructura, respaldos, etc)

- Configuración y Gestión de Cambios:
    - Identificar los elementos susceptibles de versionado
    - Definir y gestionar todas las configuraciones de dichos elementos

Flujos de Trabajo

Los flujos de trabajo indican todos los pasos que deben seguirse para dar una fase por terminada. Para las disciplinas, en cambio, los flujos de trabajo permiten asesorar a todos los involucrados para que los requerimientos sean refinados y esto permita generar la documentación precisa con los objetivos entendidos y atendidos.

De hecho, un flujo de trabajo estructurado equivocadamente puede alterar el resultado final y repercusiones en la planificación, estructuración y construcción de los casos de uso.

Un flujo de trabajo contiene elementos como actores, actividades y artefactos que interactúan entre ellos con una secuencia lógica que define claramente los requisitos para ir de una fase a otra o para dar por culminados los objetivos de una disciplina. Así pues, el nivel de detalle de cada flujo dependerá exclusivamente de los elementos que estén interactuando.

Ejemplificamos un flujo de trabajo a continuación:

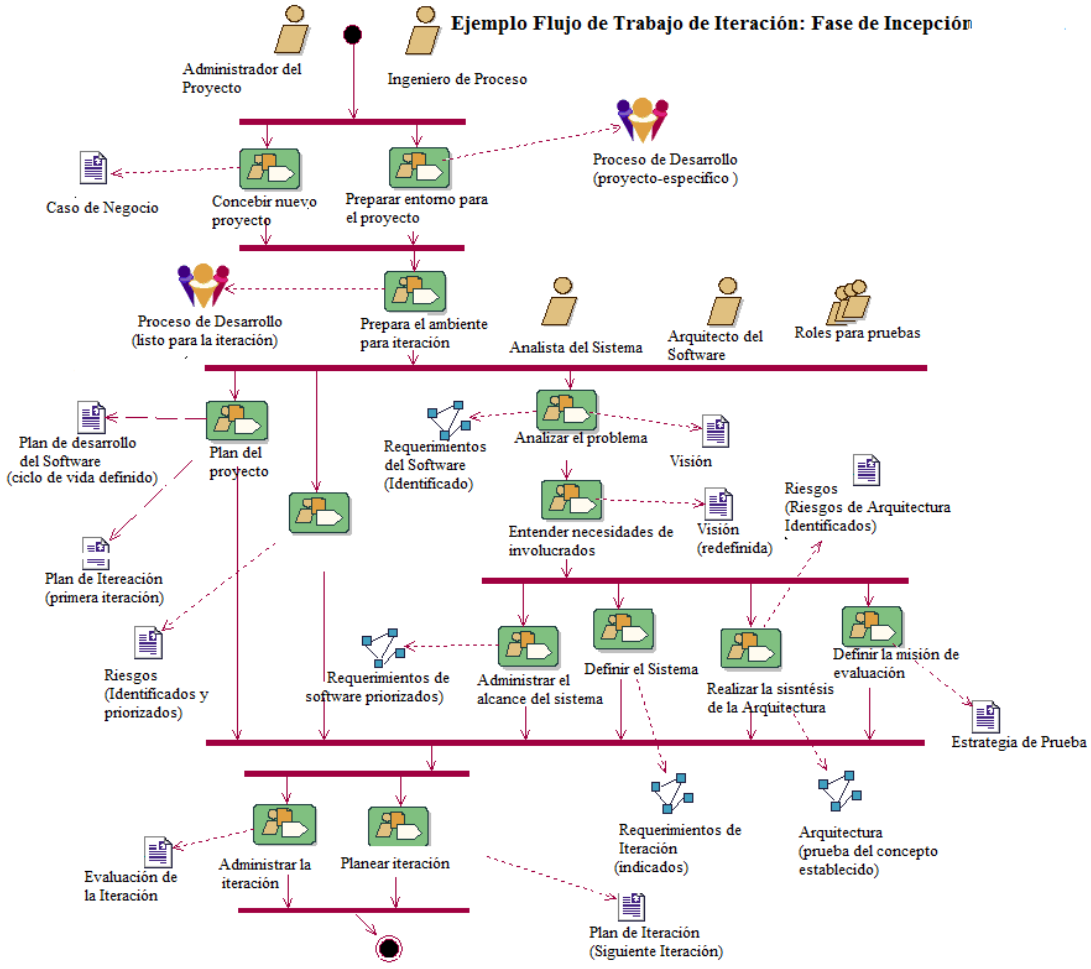


Gráfico 2.9. Ejemplo de Flujo de Trabajo en una iteración: Fase de Incepción
Editado de: Rational Software – Rational Unified Process

Del gráfico anterior podemos visualizar los actores involucrados en cada actividad contemplada para la fase de incepción; así también, vemos todos los artefactos que se van generando secuencialmente a lo largo del proceso y que denotan la formalidad de todo el trabajo realizado.

A continuación vemos un flujo de trabajo para una de las actividades en específico:

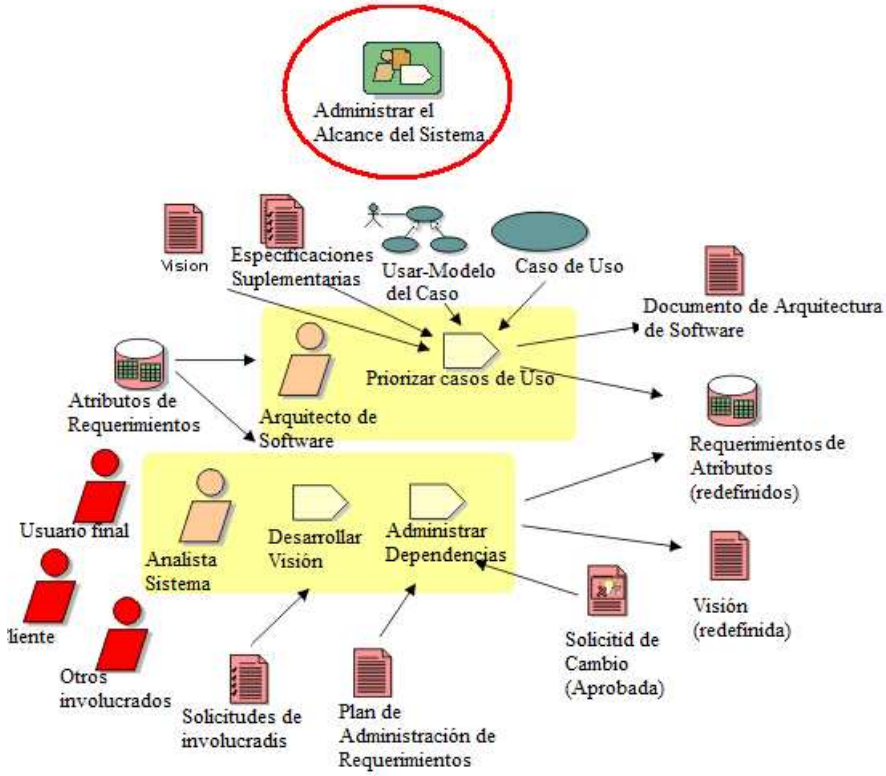


Gráfico 2.10. Flujo de Trabajo a detalle: Administrar el alcance del sistema
Editado de: Rational Software – Rational Unified Process

El gráfico anterior muestra el flujo de trabajo para la actividad 'Administrar el alcance del sistema', en el que se especifican las actividades que hay que realizar, los actores que intervienen y los artefactos que se necesitan para las actividades y, también, los artefactos que se generan luego de ejecutarlas.

Para casos en los que la persona no tiene mucha experiencia acerca de cómo desarrollar un documento o ejecutar una de las actividades, RUP proporciona una serie de pasos, objetivos y una descripción, más a detalle, de la composición de los artefactos de entrada y salida para cada actividad; en cambio para generar los artefactos, RUP suministra, también, una guía con información acerca del rol responsable de la generación, el tiempo y lapso en el que debería ser generado dentro del ciclo de vida del proyecto, la forma de adaptar el artefacto al proceso, y un par de plantillas (formal e

informal) para que el usuario no tenga complicaciones causadas por el desconocimiento.

2. **Estructura Dinámica**

La estructura dinámica de RUP permite conducir el proyecto a través de fases e iteraciones, y de objetivos a cumplir para la transición de una a otra; es decir, por un lado tenemos las disciplinas y actividades como parte de la estructura estática, y por otra parte objetivos independientes a dichos elementos que pueden incurrir en la necesidad de involucrar más de una vez a una disciplina, rol, actividad y artefactos en una iteración.

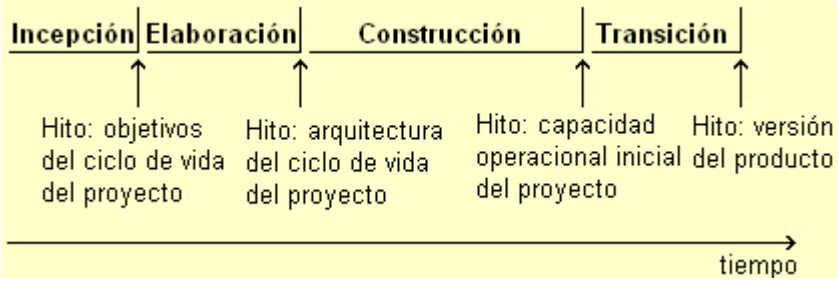


Gráfico 2.11. Vista Dinámica: Fases y Criterios de un proyecto
Editado de: Rational Software – Rational Unified Process

Las fases no son idénticas en términos de tiempo y esfuerzo. Sin embargo RUP propone que la planificación para cada fase se realice de la siguiente forma:

	Inicio	Elaboración	Construcción	Transición
% Obtenido	al menos 5%	20%	65%	10%
% Planificado	10%	30%	50%	10%

Tabla 2.1. Planificación de las fases propuesta por RUP
Editado de: Rational Software – Rational Unified Process

Así pues, queda claro que el mayor esfuerzo se debe invertir en la etapa de construcción y es allí donde transcurren la mayor cantidad de iteraciones.

Características Principales de RUP

Entre las principales características de la metodología, tenemos:

- Proceso iterativo e incremental
- Guiada por Casos de Uso
- Centrada en la Arquitectura del Software

Proceso iterativo e incremental

Como se explicó antes, un proceso es la ejecución de una serie de pasos, con un orden predeterminado, para lograr o alcanzar un meta específica. Entonces, para los procesos de ingeniería, la meta es el desarrollo de un proceso, que en RUP se ha distribuido en las distintas disciplinas que consideran flujos de trabajo, roles y artefactos.

Gráfico 2.12. Desarrollo iterativo con RUP
Editado del artículo 'Transitioning from waterfall to iterative development'

Se puede observar como cada iteración incluye actividades para requerimientos, análisis, diseño, implementación y pruebas; además, se observan las disciplinas y como las iteraciones consideradas, que representan una secuencia de actividades con un plan establecido y un

criterio de evaluación, hacen que el refinamiento de los requerimientos sea eficaz, además de finalizar con una versión del sistema. El ciclo de vida iterativo se basa en la evolución de prototipos ejecutables que se muestran a los usuarios y clientes, y en la producción de un ciclo de vida en cascada a menor escala basada en objetivos puntuales, los mismos que han sido establecidos en función de la evaluación de las iteraciones precedentes. Por tanto, en cada iteración se debe:

- Planificar la iteración y realizar el estudio de los riesgos inmersos
- Análisis de los Casos de Uso y escenarios
- Diseño de opciones arquitectónicas del software
- Realizar la codificación y las pruebas (la integración del nuevo código con el existente de iteraciones anteriores se hace gradualmente durante la construcción)
- Evaluar la entrega del producto ejecutable (evaluación del prototipo en función de las pruebas y de los criterios definidos)
- Preparar la entrega (documentación e instalación del prototipo)

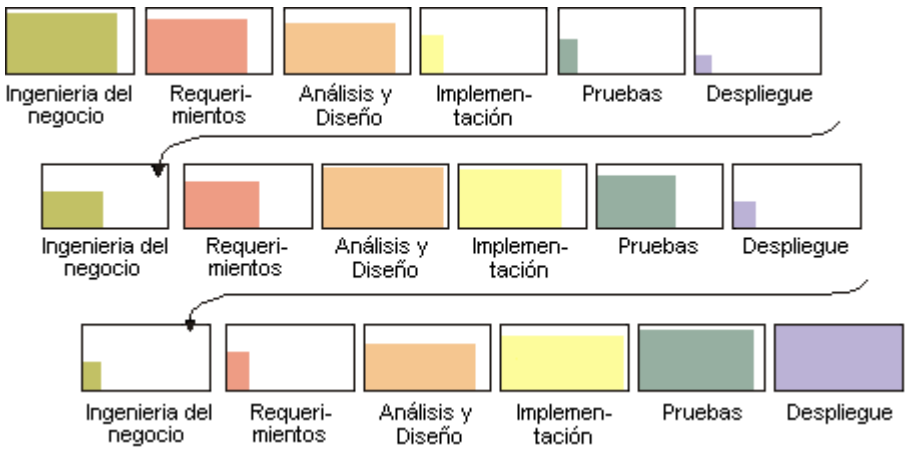


Gráfico 2.13. Enfoque Iterativo e Incremental
Editado de: http://www.ibm.com/developerworks/rational

En lo que concierne al grado de finalización de los artefactos de cada disciplina no se puede hablar de un porcentaje estándar, sin embargo es necesario que antes de empezar una nueva fase se considere la terminación de ciertos artefactos que van a ser necesarios para realizar nuevas actividades que generaran a su vez otra serie de artefactos.

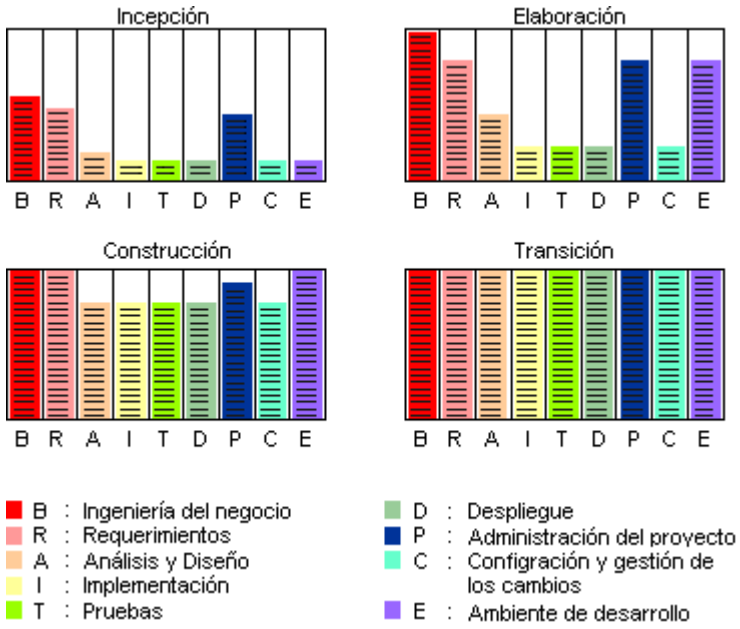


Gráfico 2.14. Grado de finalización de artefactos por disciplina y fase en RUP
Editado del artículo The IBM Rational Unified Process for System z V1.0`'

Del gráfico entonces, se observa que en cada fase se debe tener un nivel de finalización de los artefactos clasificados por disciplina. Además, y ya que en la fase de construcción es en donde pasa el mayor porcentaje del tiempo y esfuerzo del proyecto, se debe completar casi completamente todos los artefactos antes de pasar a la fase de transición.

Guiada por Casos de Uso

Los proyectos desarrollados con la metodología RUP expresan los requerimientos funcionales en forma de Casos de Uso, donde el caso es la guía de la realización de una arquitectura ejecutable para la aplicación. Además el proceso focaliza el esfuerzo del equipo en construir los elementos estructuralmente críticos, antes de construir elementos de menor impacto. Entonces, la mitigación de los riesgos más importantes guía la definición y confirmación del alcance en las primeras etapas del ciclo de vida. RUP realiza clasificaciones al ciclo de vida en iteraciones que producen versiones totalmente funcionales e incrementales de un sistema.

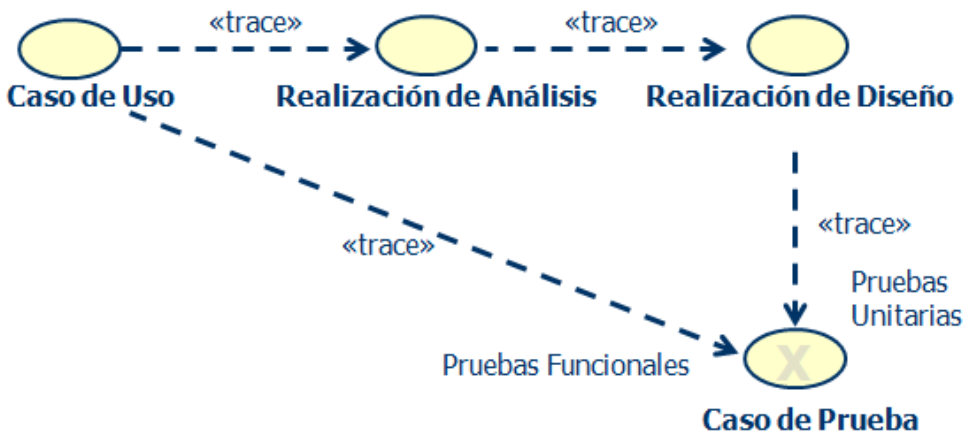


Gráfico 2.15. Funcionalidad de los Casos de Uso
Editado de: The Unified Software Development Process – I.Jacobson

Un caso de uso está totalmente atendido cuando las pruebas unitarias y funcionales han sido exitosas, es por eso que el diseño de los casos de uso debe ser adecuado, para que los casos de prueba contemplen las funcionalidades desarrolladas para el ciclo de vida en específico.

Estado de aspectos de los Casos de Uso al finalizar cada fase

	Modelo de Negocio Terminado	Casos de Uso Identificados	Casos de Uso Descritos	Casos de Uso Analizados	Casos de Uso Diseñados, Implementados y Probados
Fase de Concepción	50% - 70%	50%	10%	5%	Muy poco, puede que sólo algo relativo a un prototipo para probar conceptos
Fase de Elaboración	Casi el 100%	80% o más	40% - 80%	20% - 40%	Menos del 10%
Fase de Construcción	100%	100%	100%	100%	100%
Fase de Transición					

Gráfico 2.16. Estado de los aspectos de Casos de Uso por fase
Editado de: The Unified Software Development Process – I.Jacobson

Independientemente de la disciplina en la que se esté trabajando, RUP recomienda que la identificación, descripción, análisis, diseño, implementación y pruebas de los casos de uso debe ser gradual y proporcional a las fases; así pues, al finalizar la fase de inicio, habremos identificado al menos la mitad de los casos, descrito el décimo de ellos, analizado la mitad de los descritos y casi no haber implementado ninguno (salvo los casos en los que se necesita implementar y probar uno pequeño para conceptualizar, prolijamente, el resto). Al finalizar la fase de construcción, independientemente de si tiene una o varias iteraciones, todos los casos habrán sido desplegados.

Centrada en la Arquitectura del Software

RUP establece refinamientos sucesivos de una arquitectura ejecutable, construida como un prototipo evolutivo. La arquitectura de un sistema es la organización o estructura de sus partes más relevantes; la arquitectura ejecutable es una implementación parcial del sistema, construida para demostrar algunas funciones y propiedades.

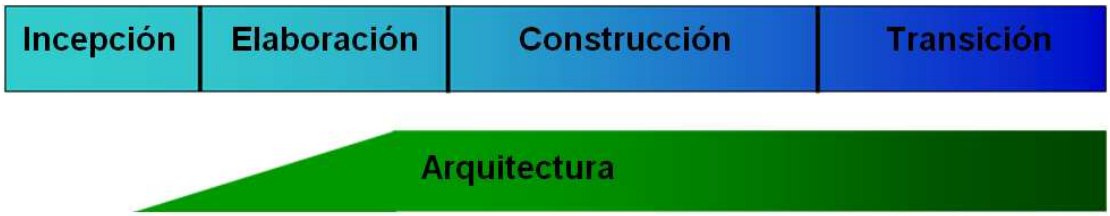


Gráfico 2.17. Evolución de la arquitectura del sistema por fase, en RUP
Editado de: The Unified Software Development Process – I.Jacobson

3. **Mejores Prácticas de RUP**

RUP ha desarrollado un enfoque que personaliza las mejores prácticas consideradas para el desarrollo de software; es decir, se contemplan los estándares internacionales para desarrollar software y, además, adhiere parámetros de calidad para la medición de los procesos. Para esto, RUP provee a cada miembro del equipo las guías de proceso, plantillas y herramientas necesarias para que se tenga ventaja de las mejores prácticas:

Gráfico 2.18. Mejores Prácticas de RUP
Editado de: http://www.histaintl.com/servicios/consulting/rup.php

Desarrollar software iterativamente: Debido a la complejidad de los sistemas de software, la misma que crece vertiginosamente en la actualidad, se vuelve nada óptimo el construirlos con estrategias que promueven el trabajo secuencial. Tenemos varios parámetros que soportan esta teoría, como: la definición y diseño completo de la solución para el requerimiento, la construcción del software y la realización de las pruebas sobre el producto resultante.

El enfoque iterativo permite tener una comprensión creciente del problema a través de refinamientos periódicos y sucesivos dentro del proceso, llegando a una solución efectiva luego de múltiples iteraciones acotadas en complejidad. RUP utiliza y soporta este enfoque iterativo generando versiones ejecutables en las que se incluyen las perspectivas del equipo de desarrollo, o proceso de aprendizaje en tiempo real, y del usuario. Es una manera acertada de mitigar los riesgos efectivamente.

Administrar los requerimientos: Los requerimientos son las condiciones o capacidades que el sistema debe conformar. La Administración de Requerimientos es un enfoque sistemático para hallar, documentar, organizar y monitorear los requerimientos cambiantes de un sistema.

La Administración de Requerimientos permite que:

a) las comunicaciones estén basadas en requerimientos claramente definidos,
b) los requerimientos sean priorizados, filtrados y monitoreados,
c) sea posible realizar evaluaciones objetivas de funcionalidad, y

d) que las inconsistencias se detecten más fácilmente.

Para esto, RUP describe como obtener, organizar y documentar la funcionalidad y restricciones requeridas, además de documentar y monitorear las alternativas y decisiones tomadas.

Utilizar arquitecturas basadas en componentes: El proceso de software debe focalizarse en el desarrollo temprano de una arquitectura robusta ejecutable, antes de comprometer recursos para el desarrollo en gran escala. RUP describe como diseñar una arquitectura flexible, que se acomode a los cambios, comprensible intuitivamente y promueve una reutilización de software más efectiva. Además, provee un enfoque sistemático para definir una arquitectura utilizando componentes nuevos y preexistentes.

Modelar software visualmente: RUP muestra como modelar software visualmente para capturar la estructura y comportamiento de arquitecturas y componentes. Las abstracciones visuales ayudan a comunicar diferentes aspectos del software, comprender los requerimientos, ver como los elementos del sistema se relacionan entre sí, mantener la consistencia entre diseño e implementación y promover una comunicación precisa. El estándar UML(Lenguaje de Modelado Unificado), creado por Rational Software, es el cimiento para un modelización visual exitosa.

Verificar la calidad de software: Es necesario evaluar la calidad de un sistema respecto de sus requerimientos de funcionalidad, confiabilidad y performance. La actividad fundamental son las pruebas, que permite

encontrar las fallas antes de la puesta en producción. RUP asiste en el planeamiento, diseño, implementación, ejecución y evaluación de todos estos tipos de prueba.

Controlar los cambios al software: La capacidad de administrar los cambios es esencial en ambientes en los cuales el cambio es inevitable. RUP describe como controlar, rastrear y monitorear los cambios para permitir un desarrollo iterativo exitoso. Es también una guía para establecer espacios de trabajo seguros para cada desarrollador, suministrando el aislamiento de los cambios hechos en otros espacios de trabajo y controlando los cambios de todos los elementos de software (modelos, código, documentos, etc.). Describe como automatizar la integración y administrar la conformación de releases.

2.2. LENGUAJES DE PROGRAMACIÓN

2.2.1. RGP

Historia

En sus orígenes el RPG era un lenguaje de programación muy simple, utilizado para generar informes en papel a partir de datos almacenados en tarjetas perforadas, desarrollado para el entorno de los sistemas 709 y 360 modelo 20 de IBM. El sistema/3 con su disco duro trajo consigo el RPG II, el cual se convirtió en un standard para el desarrollo de aplicaciones empresariales en los sistemas minis y medios de IBM. En 1960 RPG es creado para la familia 1400, pero hasta 1964 no es lanzada la versión final para la IBM 360. Ha sido actualizado en diversas ocasiones, dando origen a las diferentes versiones del lenguaje.

En la historia de los Lenguajes de Programación ha habido de todo, y RPG es un lenguaje "propietario", inventado por IBM para facilitar la programación de tareas de negocio en las Empresas. La historia del lenguaje RPG está llena de continuas mejoras y versiones, y la realidad ahora es que es la base (junto con Cobol) de los programas que funcionan en las Empresas exitosas.

En los últimos años, IBM ha mejorado en mucho RPG, ahora llamado RPG IV o RPG ILE, dotándolo de muchas opciones y funciones, mejoras en el compilador y creando el entorno ILE para facilitar la programación más estructurada y la combinación de múltiples lenguajes, como Java, C++, etc.

Descripción

El lenguaje de programación RPG (Report Program Generator) es un lenguaje de programación desarrollado por IBM y diseñado para generar informes comerciales o de negocios.

RPG está diseñado para generar los reportes de salida que resultan del proceso de aplicaciones de negocios tan comunes como son: cuentas por cobrar y cuentas por pagar. Pero el RPG puede también ser utilizado para actualizar en forma periódica los archivos de cuentas por cobrar y cuentas por pagar. Además, es un lenguaje flexible y potente un claro ejemplo es el RpgForWeb, un entorno que facilita la creación de Aplicaciones Web usando RPG IV y el estándar de la Web; html y javascript.

Desde el inicio, RPG se diseñó desde el principio como un lenguaje que utiliza la base de datos integrada en el sistema operativo. Si se quiere obtener un registro de una base de datos, basta con escribir una sentencia que lea el registro. Incluso si se quiere escribir una sentencia de SQL más compleja en el programa de RPG, se escribe la sentencia y se le indica dónde insertar o devolver datos en los nombres de variables de RPG. Conforme ha ido evolucionando el lenguaje se han desarrollado funciones incorporadas que sustituyen a muchos de los antiguos indicadores y códigos de operación. Todas estas incorporaciones permiten que el RPG se convierta en un lenguaje mucho más legible, claro, flexible y moderno. Se han desarrollado varios lenguajes pero en el entorno del mundo de los negocios los líderes (al menos en el entorno IBM) son RPG y Cobol.

RPG permite un manejo de errores adecuado. Cuando una operación de RPG falla, se envía un mensaje de escape. A menos que se escriba código para capturar y manejar el error, el programa fallará. Otros lenguajes de programación no poseen esta función. RPG ha demostrado a lo largo de todos estos años que se puede escribir un programa con él hoy en día, y dentro de cinco o diez años se podrá seguir compilando y ejecutando en el sistema.

Desde RPG se puede llamar directamente a rutinas escritas en Java o escritas en otro lenguaje ILE, con sólo crear los prototipos correctos. De esta forma, se le abrirá el mundo de las herramientas de terceros (muchas de las cuales son gratuitas) sin tener que sacrificar las ventajas del lenguaje RPG

2.2.2. COBOL

Historia

Fue diseñado por un comité en el que estaban representados los intereses del gobierno y de la industria para que fuera un lenguaje empresarial estándar, realizara cálculos correctamente, almacenara grandes cantidades de datos y los recuperase con precisión y eficacia. Las posteriores mejoras al COBOL están todas basadas en estos criterios de diseño fundamentales.

Con la llegada del Sistema Operativo Windows, son muchos los que intentan proveer al Cobol de esa interface gráfica, Objective Cobol, Visual Object Cobol de Microfocus, Fujitsu PowerCobol, Acucobol-GT, Vangui y Cobol-WOW de Liant (RM).

En contraste con otros lenguajes de programación, COBOL no se concibió para cálculos complejos matemáticos o científicos, de hecho solo dispone de comandos para realizar los cálculos mas elementales, suma, resta, multiplicación y división, sino que su empleo es apropiado para el proceso de datos en aplicaciones comerciales, utilización de grandes cantidades de datos y obtención de resultados ya sea por pantalla o impresos.

Con Cobol se pretendía un lenguaje universal, sin embargo, los numerosos fabricantes existentes en la actualidad han ido incorporando retoques y mejoras, aunque las diferencias esenciales entre ellos son mínimas.

Descripción

Cobol es un lenguaje compilado, es decir, existe el código fuente escrito con cualquier editor de textos y el código objeto (compilado) dispuesto para su ejecución con su correspondiente runtime. Cuando se realiza un programa escrito en Cobol se pueden apreciar los siguientes aspectos:

- Existen unos márgenes establecidos que facilitan su comprensión.
- Está estructurado en varias partes, cada una de ella con un objetivo dentro del programa.

COBOL solo dispone de comandos para realizar los cálculos más elementales, suma, resta, multiplicación y división, sino que su empleo es apropiado para el proceso de datos en aplicaciones comerciales, utilización de grandes cantidades de datos y obtención de resultados ya sea por pantalla o impresos.

A diferencia de otros lenguajes, COBOL se documenta a sí mismo. Hasta las personas con pocos conocimientos técnicos han aprendido COBOL en pocas semanas y lo han podido utilizar sin necesidad de entender la arquitectura interna del entorno operativo. Los usuarios sin formación en programación siempre pueden acudir al código fuente para entender la función para la que está pensado un determinado programa.

Los usuarios de COBOL pueden transportar sus aplicaciones a muchas plataformas de hardware diferentes sin necesidad de recompilar el código fuente. Esta importante característica garantiza que el hardware no se

quedará obsoleto, puesto que la inversión hecha en el software mantiene su valor.

Se puede cambiar de plataforma sin cambiar una sola línea de código. Además, esta movilidad permite que los desarrolladores que entregan programas en plataformas diferentes mantengan un solo código fuente, lo que reduce considerablemente los costes de mantenimiento.

Los programas COBOL pueden proporcionar todas las características que el usuario final desea, como el diseño y el funcionamiento Windows, la funcionalidad de seleccionar la opción deseada haciendo clic sobre ella con el ratón, la posibilidad de hacer consultas sobre datos y ejecutar informes rápidamente, y la capacidad de distribuir programas en Internet.

2.2.3. JAVA

Historia

La implementación original y de referencia del compilador, la máquina virtual y las bibliotecas de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Entre noviembre de 2006 y mayo de 2007, Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL, de acuerdo con las especificaciones del Java Community Process, de tal forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java todavía no es software libre).

Hoy en día, se puede encontrar la tecnología Java en redes y dispositivos que comprenden desde Internet y superordenadores científicos hasta portátiles y teléfonos móviles; desde simuladores de mercado en Wall Street hasta juegos de uso doméstico y tarjetas de crédito: Java está en todas partes.

Descripción

Java es un lenguaje de programación desarrollado por Sun Microsystems. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

La ejecución del código Java es segura y fiable: Los programas no acceden directamente a la memoria del ordenador, siendo imposible que un programa escrito en Java pueda acceder a los recursos del ordenador sin que esta operación le sea permitida de forma explícita. De este modo, los datos del usuario quedan a salvo de la existencia de virus escritos en Java. La ejecución segura y controlada del código Java es una característica única, que no puede encontrarse en ninguna otra tecnología.

Es totalmente multiplataforma: Es un lenguaje sencillo, por lo que el entorno necesario para su ejecución es de pequeño tamaño y puede adaptarse incluso al interior de un navegador.

Las características principales del lenguaje JAVA son las siguientes:

- Es un lenguaje orientado a objetos, esto quiere decir que se refiere a un método de programación y al diseño del lenguaje. La idea principal de orientado a objetos es diseñar el software de forma que los distintos tipos de datos que usen estén unidos a sus operaciones.

- Independencia de la plataforma, significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware.
- El recolector de basura de Java permite una fácil creación y eliminación de objetos, mayor seguridad y puede que más rápida que en el lenguaje C++. La recolección de basura de Java es un proceso prácticamente invisible al desarrollador. Es decir, el programador no tiene conciencia de cuándo la recolección de basura tendrá lugar, ya que ésta no tiene necesariamente que guardar relación con las acciones que realiza el código fuente.
- Java realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores, lo antes posible, en el ciclo de desarrollo. Java obliga a la declaración explícita de métodos, reduciendo así las posibilidades de error. Maneja la memoria para eliminar las preocupaciones por parte del programador de la liberación o corrupción de memoria.

Las funciones de JAVA son aplicadas en los siguientes entornos:

- Dispositivos móviles y sistemas empotrados
- Navegador web
- Sistemas de servidor
- Aplicaciones gráficas de escritorio

En resumen, las aplicaciones de Java resultan extremadamente seguras, ya que no acceden a zonas delicadas de memoria o de sistema, con lo cual evitan la interacción de ciertos virus. Java no posee una semántica específica para modificar la pila de programa, la memoria libre o utilizar objetos y métodos de un programa sin los privilegios del kernel del sistema operativo. Además, para evitar modificaciones por parte de los crackers de la red, implementa un método ultra seguro de autentificación por clave pública. El Cargador de Clases puede verificar una firma digital antes de realizar una instancia de un objeto.

Por tanto, ningún objeto se crea y almacena en memoria, sin que se validen los privilegios de acceso. Es decir, la seguridad se integra en el momento de compilación, con el nivel de detalle y de privilegio que sea necesario.

2.2.4. LOTUS NOTES

Historia

La versión original de Lotus Notes se llamó PLATO Notes realizada en 1973 en el CERL (Computer-based Education Research Laboratory) en la universidad de Illinois, y ya contemplaba la comunicación segura entre los usuarios. Psteriormente Tim Halvorsen, Len Kawell y Steven Beckhardt se incorporaron al proyecto. Todos ellos aportaron sus conocimientos para crear un nuevo software de colaboración y mensajería. Ellos modelaron Lotus Notes a partir de PLATO Notes pero expandiendo el concepto con un sinfín de nuevas posibilidades. Alan Eldridge, de Digital Equipment Corporation, contribuyó con el sistema de bases de datos y seguridad en la arquitectura de Lotus Notes. La visión original de Lotus Notes integraba grupos de discusión, e-mail, libretas de direcciones y bases de datos documentales. Este concepto, pero, tenia 2 problemas: el tipo de redes necesarias todavía estaba en una edad muy temprana y los sistemas operativos de los PCs eran muy inmaduros para soportar dicha idea. Para resolver estos inconvenientes tuvieron que desarrollar un servidor de nombres, bases de datos y redes para poder llevar a cabo su idea.

Lotus Notes tuvo un éxito inicial impresionante. Los directivos de Price Waterhouse vieron una demostración de la primera versión de producto y quedaron impresionados. Se vendieron 10.000 copias. Price Waterhouse predijo que Lotus Notes transformaría la forma de trabajar en las empresas. Notes salió a la venta en 1998; su configuración original solo ofrecía el correo electrónico y los documentos en grupo como funciones estándar. La clave del

éxito de Notes fue haber sido el primer producto de groupware y haber contado con el respaldo de Lotus y de IBM.

Descripción

Lotus Notes es un sistema cliente - servidor propietario de trabajo colaborativo y correo electrónico, desarrollado por Lotus Software, filial de IBM.

El servidor Lotus Domino dispone de versiones para distintas plataformas, incluyendo Windows NT, Windows 2000, Linux, HP-UX y Solaris, i5OS y z/OS.

El cliente Lotus Notes dispone de versiones nativas para Windows y Mac OS, y para Linux.

Lotus Domino/ Notes es un sistema de comunicación el cual permite enviar correo electrónico y manejo de Calendarios y Agendas. También es una plataforma de colaboración que permite compartir bases de datos con información como por ejemplo, bases documentales, de procedimientos, manuales o foros de discusión.

Lotus Notes es un entorno abierto para el uso compartido de conocimientos y para la creación de aplicaciones vitales en el mundo empresarial. Su núcleo está constituido por una exclusiva tecnología de gestión de documentos y de objetos, que se ocupa de almacenar la información de manera segura en bases de datos compartidas situadas en servidores de red, y que facilita a toda una organización el acceso rápido a la información.

Las principales características del lenguaje Lotus Notes son las siguientes:

- Lotus ofrece alta disponibilidad mediante un Domino Clustering, el cual es a nivel Domino por lo que permite tener hasta 6 nodos en una mezcla de sistemas operativos, ejemplo Nodo1 (con Domino sobre Windows) haciendo clúster local con un Nodo 2 (con Domino sobre Linux) y con un Nodo 3 remoto (con Domino sobre Sun Solaris o cualquier otro sistema operativo soportado). El clúster de Domino permite el balanceo de cargas y/o "failover", para clientes Notes y clientes web.
- Permite a grupos de personas tener acceso, rastrear, actualizar, organizar y compartir información; además de que les permite participar en discusiones de grupo, distribuir informes y administrar proyectos.
- Es utilizado como motor de aplicaciones de todo tipo, desde las tareas habituales de coordinación de grupos de trabajo, pasando por las publicaciones electrónicas, administración de sistemas de fax, gestión de llamadas telefónicas y del correo electrónico y tratamiento de documentos, hasta la gestión de proyectos: prácticamente toda la gestión de información dentro de la empresa puede ser abordada por Notes.
- Una de las funciones más importantes en el cliente de Lotus Notes es la inclusión de la suite de productividad Symphony, que se basa en las aplicaciones de productividad de OpenOffice.org. Esto hace posible acceder a herramientas de procesamiento de texto, hoja de cálculo y presentaciones directamente desde dentro del cliente de Notes.

- Acceso a los principales datos de una empresa, sea cual sea su ubicación. Con Notes es posible intercambiar datos con sistemas de bases de datos relacionales y de transacción, incluyendo extensiones de LotusScript para ODBC, MQSeries y muchos otros.
- Es posible crear aplicaciones y documentos Notes más complejos mediante eficaces programas laborales basados en ActiveX.

En conclusión Lotus Notes ha sido una aplicación distribuida desde su origen y funciona de forma adecuada en la mayoría de los casos. No obstante, la naturaleza de la duplicación (o replicación) de bases de datos constante y minuciosa de Notes genera una gran cantidad de tráfico WAN para los usuarios remotos. Aunque el protocolo de aplicación de Notes es bastante eficaz, el comportamiento de TCP combinado con un tráfico alto provocan que la duplicación se ralentice en extremo en entornos de ancho de banda reducido y alta latencia.

CAPITULO III

SITUACIÓN ACTUAL

3.1. APLICACIONES DESARROLLADAS EN PETROCOMERCIAL

3.1.1. RPG

- **HISTORIA Y DESCRIPCIÓN GENERAL DEL LENGUAJE**

*R*eport *P*rogram *G*enerator, como son sus siglas en inglés, es un lenguaje "propietario" inventado por IBM en la década de los sesentas para facilitar la programación de tareas de negocio. Fue diseñado para generar informes comerciales o de negocios.

En los últimos años, hasta el 2001, IBM ha mejorado en mucho a RPG, dotándolo de muchas opciones, funciones, mejoras en el compilador y creando el entorno ILE para facilitar la programación más estructurada, así como la combinación de múltiples lenguajes como Java, C++, etc.

- **ESTRUCTURA Y CONCEPTUALIZACIÓN DE LOS SISTEMAS DESARROLLADOS EN RPG**

Los sistemas desarrollados en RPG son:

- *SISTEMA DE RECURSOS HUMANOS*

El Sistema de Recursos Humanos es un sistema dinámico que está en constante actualización y ajuste debido a los distintos requerimientos que se suscitan, como son:

- Leyes y reglamentos dictados por los contratos colectivos

- Directrices definidos por la Asamblea Constituyente
- Leyes definidas por el Ministerio de Trabajo

Todos los cambios, modificaciones e implementaciones al Sistema de recursos Humanos son de tipo funcional, y deben ser ajustados al requerimiento solicitado de manera rápida, eficaz. Por ejemplo, a través del Contrato Colectivo se informa que algún beneficio dio por culminada su vigencia, por lo que se procede a eliminarlo de la base de cálculos del sistema.

Los proyectos de desarrollo de RPG son definidos por el Departamento de Recursos Humanos y siempre hacen referencia a la funcionalidad del sistema; sin embargo, existen casos en los que los cambios son definidos en las actas de reuniones de los organismos de control externos que mantienen reuniones para tratar y definir normativas generales. Cuando estos casos se suscitan, se envía un comunicado al Ing. Cristóbal Salazar, administrador de la aplicación, directamente y se indica el cambio que debe ser aplicado.

- **ELEMENTOS QUE INTERVIENEN EN EL PROCESO DE DESARROLLO DE SISTEMAS EN RPG**

1. FASES

- **ANÁLISIS**

En esta fase, el personal de PETROCOMERCIAL realiza la concepción del proyecto de desarrollo mediante el análisis del requerimiento o acatando las distintas disposiciones de los organismos de control. Para esto, se identifica claramente el objeto de la creación, modificación o eliminación, y se lleva una bitácora manual del análisis realizado en la que se registra la adhesión, eliminación o algún punto destacable del requerimiento en cuestión.

El formato en el que se presenta un requerimiento puede venir en distintos formatos:

- Fotocopia del requerimiento, sumillada, con la información original (se evita la utilización de un memorándum debido a que implicaría un paso burocrático que no se puede dar por la brevedad con la que se deben implementar los cambios)

- Correo electrónico, en caso de cambios menores, como actas de reuniones con una sumilla de “Sistemas aplicar, en el cual se decide que firmen los responsables.

En el caso que el impacto a la base de datos sea significativo, se sostiene una conversación con el personal de las Jefaturas de cada departamento involucrado y con la administradora de la base de datos para que brinde una

guía para refinar el requerimiento, si fuese necesario. Como consecuencia de esta reunión se tiene un estudio de la administración de los recursos tecnológicos como el espacio que se designará para el producto de software en los servidores.

Los requerimientos pueden impactar de las siguientes formas:

Tipos de Impacto de un requerimiento	Descripción
Modificación de vistas	Sentencias lógicas SQL que aglutinan información, la unifican y presentan para que sea visualizada de distintas maneras
Cambio a las pantallas de presentaciones	Modificaciones en la forma de la presentación de la información en algún menú del sistema
Cambios de programación	Modificaciones en la sincronización o secuencialidad de los códigos programados
Cambios a la programación	Adición, modificación o eliminación de códigos programados para incrementar, cambiar o mermar las funcionalidades propias del sistema

Tabla 3.1. Descripción de los tipos de impacto de un requerimiento
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **DISEÑO**

Actualmente, en esta etapa se establece la forma en la que el sistema cumplirá con los requerimientos identificados durante la fase de análisis; se identificarán las salidas que debe producir el sistema, y dependiendo de la criticidad del requerimiento se decide el tiempo que tomará el cumplimiento del mismo.

Para el caso en el que se tiene una disposición que indica la eliminación de algún beneficio del contrato colectivo, se debe proceder inmediatamente y la fase de análisis y diseño se fusionan con la de desarrollo y pruebas.

En el diseño, también se especifican los datos de entrada, los datos que serán base para los cálculos y los datos que deben ser almacenados, esto colabora para que la selección de las estructuras de archivo y los dispositivos de almacenamiento sea más efectiva. En ocasiones, la información detallada del diseño se registra en .la misma bitácora, pero siempre es utilizada para dar comienzo a la fase de desarrollo de software.

- **DESARROLLO**

El administrador de la aplicación realiza la programación y la documentación de los sistemas. Adicionalmente, proporciona una explicación del cómo y porqué de la codificación específica en ciertos procedimientos. La documentación es esencial para probar el programa y llevar a cabo el mantenimiento una vez que la aplicación se encuentra instalada.

La metodología que se sigue para desarrollar es secuencial, donde se realizan pequeñas rutinas o procedimientos, se prueban los mismos, para luego pasar a la fase de pruebas globales. Cabe destacar que no se genera documentación alguna de las pruebas de desarrollo.

- **PRUEBAS**

Se tiene un ambiente de pruebas que es utilizado, mayormente, en las ocasiones en que los desarrollos son extensos y complejos, y se necesita de períodos más extensos de pruebas. En dicho ambiente se tiene registrada información con datos actuales, los mismos que pueden ser manipulados para medir la eficacia de los procedimientos. No se genera documentación alguna del resultado de las pruebas y de la composición de las pruebas.

- **IMPLEMENTACIÓN, EVALUACIÓN Y MANTENIMIENTO**

Luego de realizar las pruebas sobre los procedimientos del sistema desarrollado, se procede a referenciar el proyecto al ambiente de producción; para esto se re direcciona a las librerías correspondientes, se compila, y el usuario identifica o evalúa la efectividad de los procedimientos del requerimiento cumplido. En caso que exista de una implementación no aceptada, se reporta al administrador de la aplicación para que proceda con la realización de las correcciones pertinentes.

El mantenimiento de la aplicación, sistema o producto se lo realiza mediante la comprensión de las facetas y procesos inmersos en el desarrollo. Este conocimiento implica, y busca dar respuestas, a las siguientes interrogantes: ¿Qué es lo que se hace?, ¿Cómo se hace?, ¿Con qué frecuencia se presenta?, ¿Cuál es el grado de eficiencia con el que se efectúan las tareas?, ¿Existe algún problema?; Si existe un problema, ¿qué tan serio es? ó ¿Cuál es la causa que lo originó?. El Departamento de Recursos Humanos y el

administrador de la aplicación deben retroalimentar su conocimiento con los hallazgos.

2. ROLES, CARGOS Y PERSONAS

Del relevamiento realizado se nos informó que las personas que intervienen en el proceso de desarrollo son:

Nombre	Cargo	Rol	Fase
Ing. Cristóbal Salazar	Especialista de Sistemas IIIA	Jefe del Proyecto	Todas
		Analista de Sistemas	Análisis, Diseño
		Arquitecto de Software	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador	Desarrollo
		Integrador	Desarrollo, Pruebas
		Implementador	Implementación
Usuario	Personal del Área Usuaria	Verificador (persona que realizará las pruebas)	Pruebas

Tabla 3.2. Descripción de los roles, cargos y personas que intervienen en el proceso de desarrollo de software con RPG
Realizado por: Xavier Salazar / Eduardo Velalcázar

Además de las personas que constan en la tabla, intervienen los Jefes de Área que vienen a ser supervisores del proceso de aprobación. Por ejemplo, si un usuario del Departamento de Recursos Humanos identifica un requerimiento, el memorándum que hace referencia a dicho requerimiento es aprobado y redactado por el Jefe de Área, y dirigido al Jefe de Sistemas, para que sea él quien avale cualquier característica del mismo. Como se especificó en el relevamiento, la Administradora de Base de Datos interviene en los casos en los requerimientos tienen gran impacto; sin embargo, su intervención es como guía y consulta para refinar el requerimiento, mas no de aprobación netamente.

3.1.1.1. DIAGRAMAS DE FLUJO DEL PROCESO DE DESARROLLO

a) Proceso General de Desarrollo de Software

- **Diagrama de Flujo**

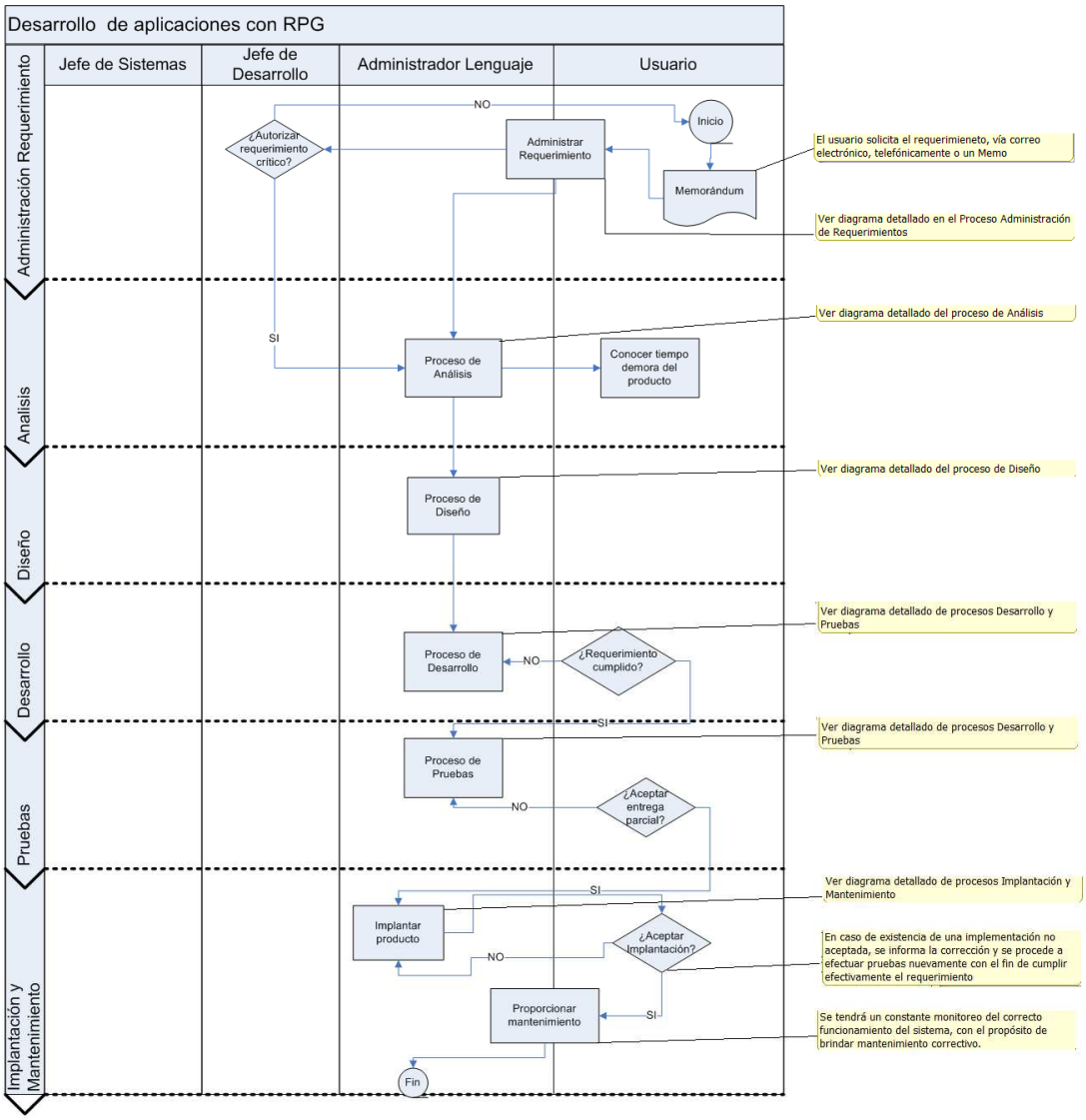
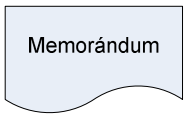
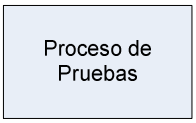


Gráfico 3.1. RPG: Proceso General de Desarrollo de Software
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de fases, elementos y responsables**

Elemento	Descripción	Responsable	Fase
	Documento en el que se registra la información más importante del requerimiento.	Área Usuaria	Requerimientos
	El administrador del Sistema Recursos Humanos registra los cambios solicitados en una bitácora manual. El usuario lleva a cabo una reunión con el encargado de desarrollar lo requerido, con el motivo de entender lo solicitado. El administrador del Sistema Recursos Humanos registra los cambios solicitados en una bitácora manual.	Área Usuaria / Administrador aplicación	
	Revisar los factibles cambios que pueden ser afectados tanto en la estructura de la base de datos como, en la base de cálculos. No se lleva un cronograma de todas las actividades a realizarse, por lo tanto el administrador de la aplicación en base a su experiencia define un tiempo aproximado a cumplirse el requerimiento Identificar si se debe crear, modificar o eliminar en la aplicación ya sean tablas o campos en la base de datos.	Área Usuaria / Administrador aplicación	Análisis
	Dependiendo de la criticidad del requerimiento, se registra un tiempo adecuado.	Administrador aplicación	Diseño
	Iniciar la programación a través del lenguaje RPG 400 y se decide de emplear o no el ambiente de pruebas.	Administrador aplicación	Desarrollo
	En caso de un cambio en la base de cálculo se utiliza el ambiente de pruebas, se respalda la información y se efectúan los cambios que se reflejarán luego en el ambiente de producción.	Administrador aplicación	Pruebas

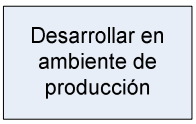
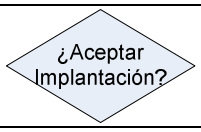
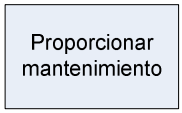
	En caso de cambios poco significativos, que no afecten la integridad de la base de datos, se procederá a desarrollar en la base de cálculo.	Administrador aplicación	Desarrollo
	Si el usuario se encuentra conforme con el avance entregado, prosigue la programación previamente planificada.	Área Usuaria / Administrador aplicación	Pruebas
	Al culminar la etapa de pruebas, se copian las aplicaciones en el ambiente de producción. Al compilar este proceso satisfactoriamente entonces el programa está listo para ser ejecutado.	Administrador aplicación	Implantación
	El usuario decide si está conforme con la implantación efectuada ya culminada.	Área Usuaria	
	Se tendrá un constante monitoreo del correcto funcionamiento del sistema, con el propósito de brindar mantenimiento correctivo.	Administrador aplicación	Mantenimiento

Tabla 3.3. RPG: Proceso General de Desarrollo de Software
Descripción de fases, elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

b) Diagramas detallados en cada fase

Fase: ADMINISTRACIÓN DE REQUERIMIENTOS

Proceso: Administrar Requerimiento

- **Diagrama de flujo**

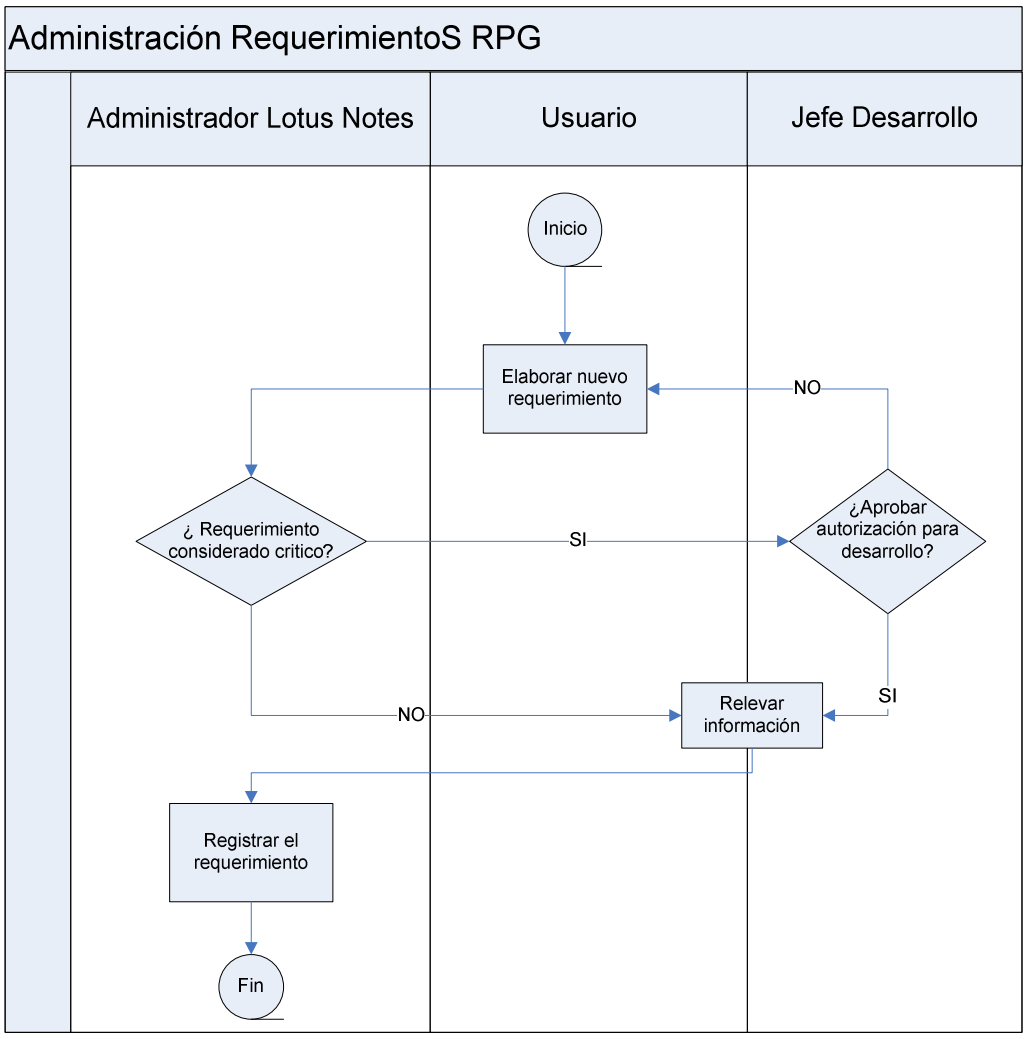


Gráfico 3.2. RPG: Proceso Administrar Requerimiento
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

Elemento	Descripción	Responsable
Elaborar nuevo requerimiento	Los requerimientos son elaborados por el Departamento de RR.HH y solicitados por la Asamblea Constituyente, el Ministerio de Trabajo, por Ajustes de código SENRES (Secretaria Nacional Técnica de Desarrollo de Recursos Humanos y Remuneración del Sector Público), el Ministerio de Finanzas, la DNH (Dirección Nacional de Hidrocarburos).	Área Usuaria
¿Aprobar autorización para desarrollo?	En caso de que se suscite un requerimiento crítico, es necesaria la aprobación por parte de la jefatura de desarrollo.	Jefa de Desarrollo
Relevar información	El usuario lleva a cabo una reunión con el encargado de desarrollar lo requerido, con el motivo de entender lo solicitado.	Área Usuaria / Administrador aplicación
Registrar el requerimiento	El administrador del Sistema Recursos Humanos registra los cambios solicitados en una bitácora manual.	Administrador aplicación

Tabla 3.4. RPG: Proceso Registro Requerimientos
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fase y Proceso: ANÁLISIS

- **Diagrama de Flujo**

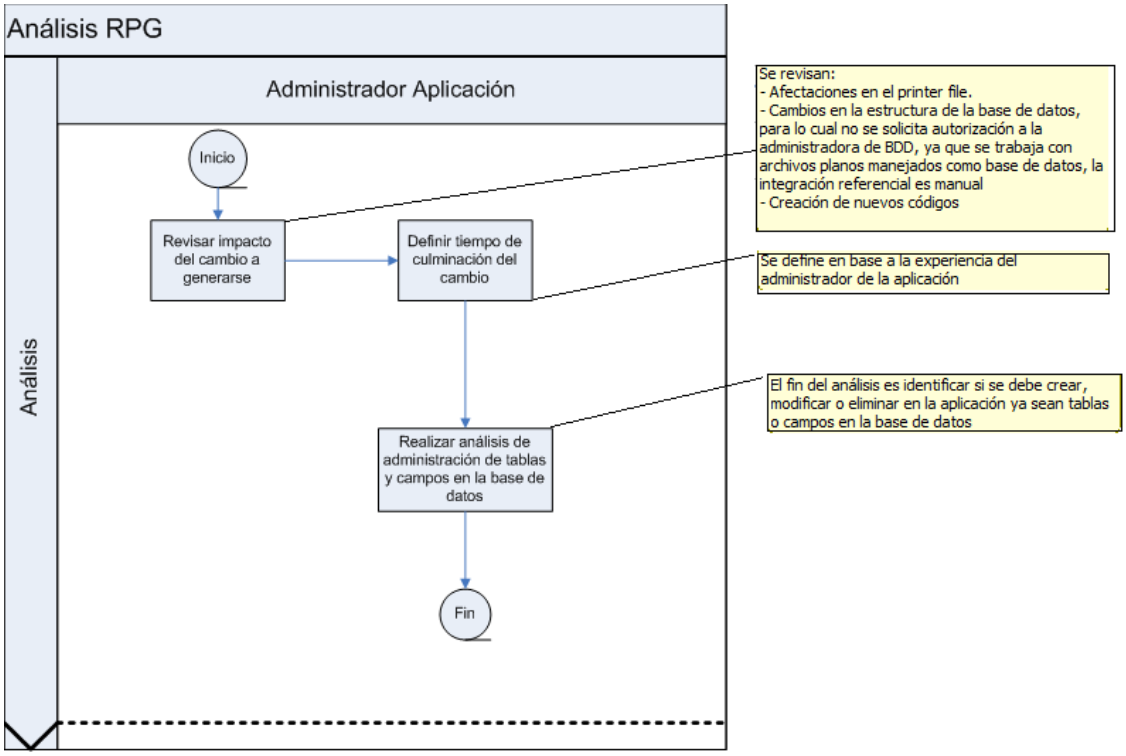


Gráfico 3.3. RPG: Proceso Análisis
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

Elemento	Descripción	Responsables
Revisar impacto del cambio a generarse	Revisar los factibles cambios que pueden ser afectados tanto en la estructura de la base de datos como, en la base de cálculos.	Administrador aplicación
Definir tiempo de culminación del cambio	No se lleva un cronograma de las actividades a realizarse, por lo tanto el administrador de la aplicación en base a su experiencia define un tiempo aproximado a cumplirse el requerimiento.	Administrador aplicación
Realizar análisis de administración de tablas y campos en la base de datos	El fin del análisis es identificar si se debe crear, modificar o eliminar en la aplicación ya sean tablas o campos en la base de datos.	Administrador aplicación

Tabla 3.5. RPG: Proceso de Análisis
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fase y Proceso: DISEÑO

- **Diagrama de Flujo**

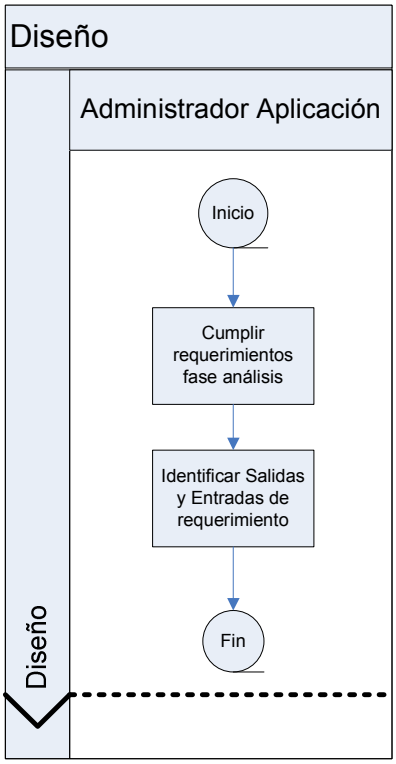


Gráfico 3.4. RPG: Proceso de Diseño
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

Elemento	Descripción	Responsable
Cumplir requerimientos fase análisis	Se analizan las características del requerimiento inicial y se diseña la solución para que cumpla con las especificaciones. Si la criticidad es importante se amplía el tiempo que tomará la realización de la solución.	Administrador aplicación
Identificar Salidas y Entradas de requerimiento	No se lleva un cronograma de las actividades a realizarse, por lo tanto el administrador de la aplicación en base a su experiencia define un tiempo aproximado a cumplirse el requerimiento.	Administrador aplicación

Tabla 3.6. RPG: Proceso Diseño
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fases y Procesos: DESARROLLO y PRUEBAS

- **Diagrama de Flujo**

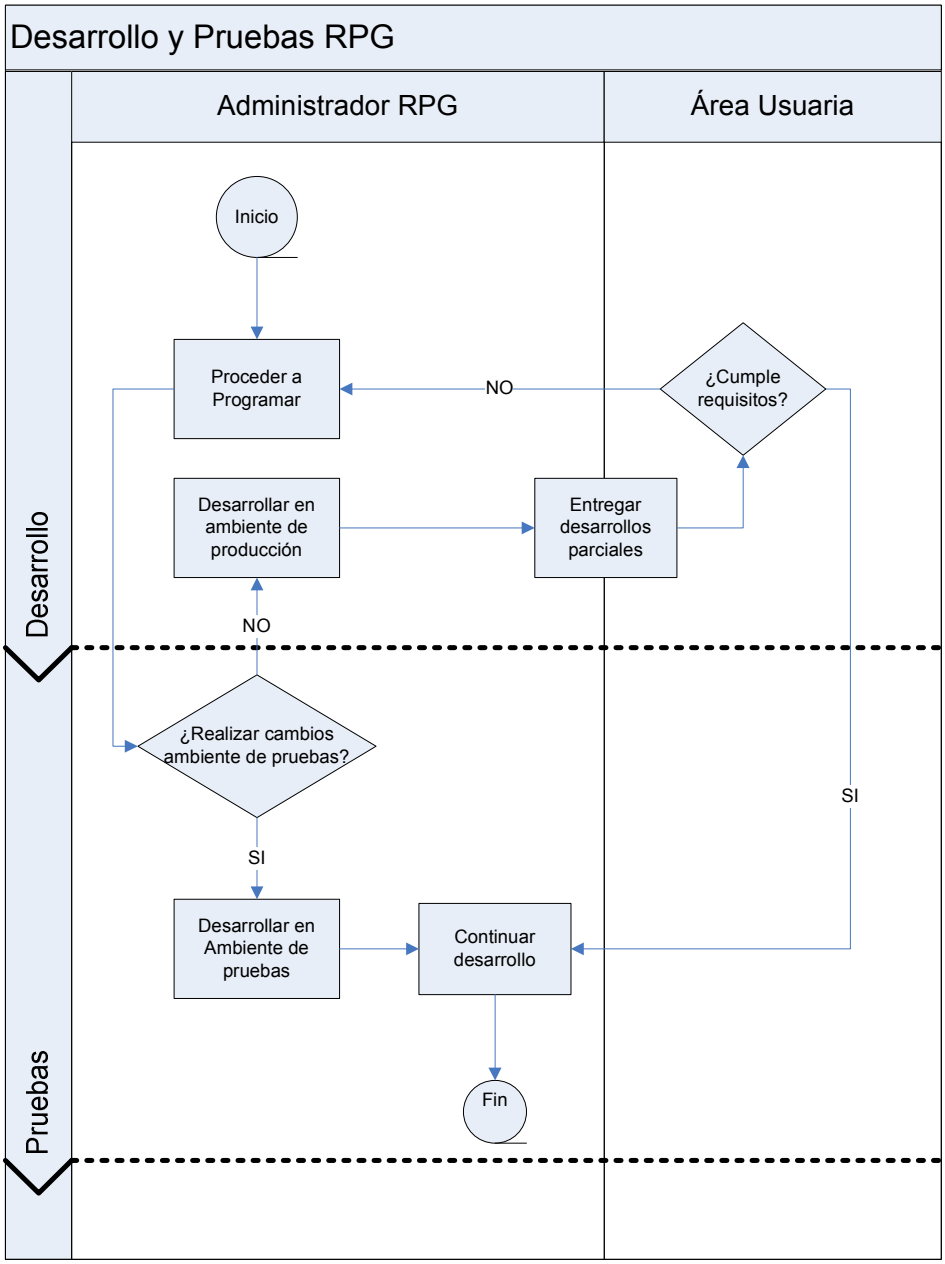


Gráfico 3.5. RPG: Procesos de Desarrollo y Pruebas
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de fases, elementos y responsables**

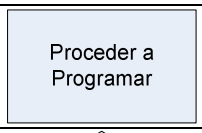
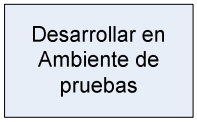
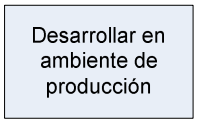
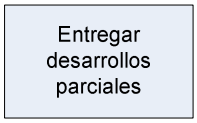
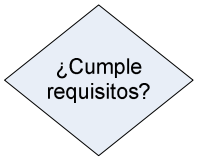
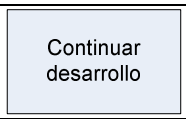
Elemento	Descripción	Responsable	Fase
	Iniciar la programación a través del lenguaje RPG 400.	Administrador aplicación	Desarrollo
	Se toma la decisión de emplear o no el ambiente de pruebas.	Administrador aplicación	Pruebas
	En caso de un cambio en la base de cálculo, se utiliza el ambiente de pruebas, se respalda la información y se efectúan los cambios que afectarán en el ambiente de producción.	Administrador aplicación	Desarrollo
	En caso de cambios poco significativos, que no afecten la integridad de la base de datos, se procederá a desarrollar en la base de cálculo.	Administrador aplicación	Desarrollo
	Se realizan entregas parciales conforme avanza el desarrollo de los aplicativos, el área usuaria tiene la responsabilidad de cargar la información correspondiente al sistema.	Área Usuaria / Administrador aplicación	Desarrollo
	Si el usuario lo aprueba el avance entregado, se prosigue con la programación que se planificó previamente.	Área Usuaria	Desarrollo
	El proceso de desarrollo pasa a la siguiente actividad.	Administrador aplicación	Pruebas

Tabla 3.7. RPG: Procesos Desarrollo y Pruebas
Descripción de fases, elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fases: IMPLANTACIÓN **MANTENIMIENTO**

Procesos: Implantar producto **Proporcionar mantenimiento**

- **Diagrama de Flujo**

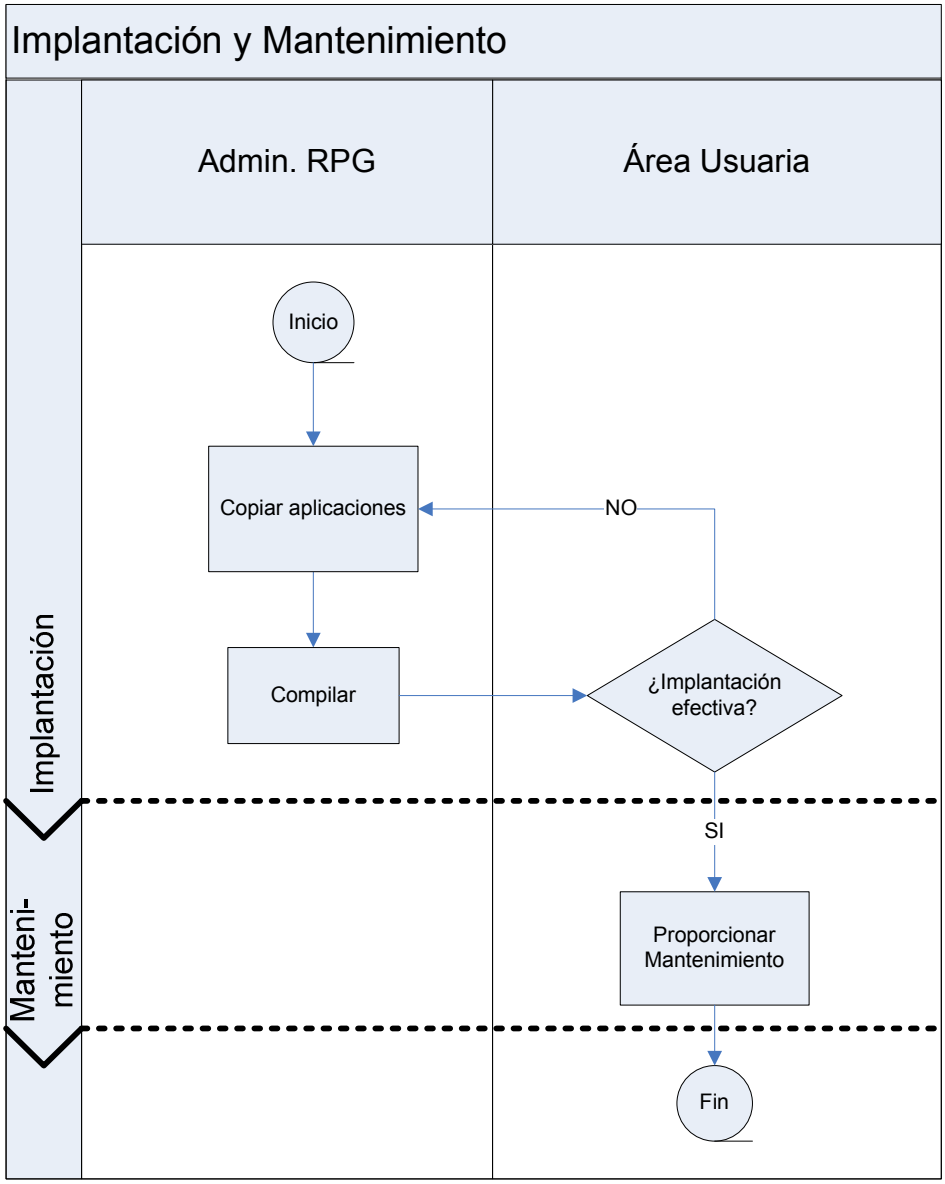


Gráfico 3.6. RPG: Procesos Implantación y Mantenimiento
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de fases, elementos y responsables**

Proceso	Objetivo	Responsable	Fase
Copiar aplicaciones	Al culminar la etapa de pruebas, se copian las aplicaciones en el ambiente de producción.	Administrador aplicación	Implantación
Compilar	Al compilar este proceso satisfactoriamente entonces el programa está listo para ser ejecutado.	Administrador aplicación	
¿Implantación efectiva?	El usuario decide si está conforme con la implantación efectuada y culminada.	Área Usuaria	
Proporcionar mantenimiento	Se tendrá un constante monitoreo del correcto funcionamiento del sistema, con el propósito de brindar mantenimiento correctivo.	Administrador aplicación	Mantenimiento

Tabla 3.8. RPG: Procesos Implantación y Mantenimiento
Descripción de fases, elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

3.1.2. COBOL

- **HISTORIA Y DESCRIPCIÓN GENERAL DEL LENGUAJE**

*CO*mmon *B*usiness - *O*riented *L*anguage fue creado en el año 1960 con el objetivo de crear un lenguaje de programación universal que pudiera ser usado en cualquier ordenador, ya que para la época se tenían numerosos modelos de ordenadores incompatibles entre sí, y que estuviera orientado principalmente a los negocios, es decir, a la llamada informática de gestión.

A partir de aquí, apoyado por las sugerencias de los usuarios y expertos, su evolución no paró hasta que en 1968 y posteriormente en 1974, se dieron las primeras versiones consideradas estándar, que fueron conocidas como Cobol Ansi.

Al principio, los mainframes eran los tipos de ordenador escogidos por las grandes empresas, sobre todo las del sector financiero, con sus propios sistemas operativos y sus propios compiladores de Cobol. Estos fueron afianzando al Cobol como un lenguaje perfecto para conseguir sus propósitos por su robustez, su fiabilidad y su perfecta adaptación a las necesidades de gestión.

Cuando los sistemas operativos empezaron a independizarse de las máquinas, fue cuando los fabricantes de compiladores Cobol comenzaron su expansión. Así pues, empresas como Liant, Acucorp, Merant, Fujitsu, Nigsun, IBM o Computer Associates, dan cabida a la continuidad en la programación con éste lenguaje

Cobol es un lenguaje compilado. Es decir, consta de un código fuente perfectamente legible y adaptado a unas normas, que se puede realizar con cualquier editor de textos y un código objeto (compilado) dispuesto para su ejecución con su correspondiente runtime.

Características de COBOL:

- Existen unos márgenes establecidos que facilitan su comprensión.
- Está estructurado en cuatro partes, cada una de ella con un objetivo dentro del programa.
- La gramática y su vocabulario tienen su base en la lengua inglesa.
- En contraste con otros lenguajes de programación, Cobol no fue concebido para cálculos complejos matemáticos o científicos (de hecho solo dispone de comandos para realizar los cálculos más elementales), aunque si posee un elevado grado de precisión y velocidad del cálculo numérico, pudiendo manejar hasta 30 posiciones decimales.
- Aunque no sea el único lenguaje orientado a éste propósito, si ha sido el más utilizado en toda la historia.
- Lenguaje independiente de la plataforma en la que se ejecute, por lo tanto es posible ejecutar el mismo programa sin modificar nada en cientos de sistemas diferentes (Windows, Unix, MS-Dos, Linux, OS400, S36, S34, VMS, Netware, Solaris, etc...)

- **ESTRUCTURA Y CONCEPTUALIZACIÓN DE LOS SISTEMAS DESARROLLADOS EN COBOL**

Los sistemas desarrollados en COBOL son:

- *SISTEMA DE MOVIMIENTO DE PRODUCTO*

El Sistema de Movimiento De Producto es un sistema que se encarga de la gestión de los productos que produce PETROCOMERCIAL. La gestión consiste en administrar la recepción, almacenamiento y despacho del producto a nivel nacional.

La configuración de los parámetros y directrices que dan forma al sistema son concebidas, identificadas o solicitadas por los distintos departamentos inmersos en el proceso de movimiento de producto. Algunos de estos departamentos son sistemas, programación, operaciones. Además de estos departamentos, los requerimientos también pueden suscitarse por regulaciones externas dictadas por organismos de control o el gobierno.

El sistema fue implementado hace aproximadamente 20 años y está instalado en las dependencias que funcionan en todo el país.

Al presentarse cualquier requerimiento para realizar alguna implementación importante en el sistema, la persona que viabiliza las fases del desarrollo de la solución es el Ing. Julio Delgado, quien es el administrador de la aplicación.

- **ELEMENTOS QUE INTERVIENEN EN EL PROCESO DE DESARROLLO DE SISTEMAS EN COBOL**

1. FASES

- **ANÁLISIS**

En esta fase, el personal de PETROCOMERCIAL realiza la concepción del requerimiento. Para esto, el área usuaria identifica el requerimiento, el mismo que viene en formato Memo, y es enviada de una Jefatura Departamental a otra, o en caso de que sea el mismo departamento, el meo es enviado por el Jefe de Área al Jefe del Departamento. Y si se estima necesario informar a más un Departamento, entonces se redacta otro memo informativo, más no con un reporte del requerimiento encontrado. Para casos muy especiales o de menor impacto, la forma de comunicar el requerimiento es vía telefónica o en un correo electrónico.

Luego, el administrador de la aplicación realiza un análisis del requerimiento y lo clasifica, con el fin de saber si se trata de: cambio a la base de datos (Ej: el producto X o el servicio Y está asociado a un código 1, pero el usuario solicita y quiero el 2. O a su vez, se tiene la necesidad de crear un código nuevo), implementación funcional (Ej: un módulo para la administración de los productos incautados), ó requerimiento ya implementado (Ej: En MOPRO se ha implementado una guía de remisión pero el usuario no sabe cómo utilizarla).

Las personas que intervienen en este proceso son el Administrador de la Aplicación y el usuario que requiere o el área que solicita.

- **DISEÑO**

Como resultado del análisis, se procede con el diseño del requerimiento. Este diseño es informal y se lo realiza en papeles o en una pizarra. El motivo por el que esta acción tiene lugar, es para que el área usuaria visualice como se procesará la información, donde se van a reflejar los cambios a implementar y, también, cómo va a lucir el módulo. Luego, se hace la planificación del tiempo que tomará el desarrollo y la implementación del requerimiento en la misma reunión. Se llega a un acuerdo verbal con el cliente para que éste conozca de los plazos que parametrizan el desarrollo; y, al final, se registra dicha planificación en un acta.

- **DESARROLLO**

Se procede con la etapa de desarrollo, en la que el Administrador de la aplicación se encarga de tomar la información relevada y a partir de allí decide qué módulo o sección debe implementarse primero; es decir, prioriza los procesos para optimizar el desarrollo. Se desarrolla de la siguiente forma:

- Primero se realiza el modelo de base de datos. Esto hace mención a la realización de la constitución misma de las tablas como los atributos y funciones.
- Luego, se diseñan pantallas que permitan ingresar la información para realizar pruebas de desarrollo en las que se busca validar los datos y procesos para evaluar su consistencia.
- El área usuaria o el administrador de la aplicación ingresa la información necesaria.
- El administrador genera el código fuente.

- Finalmente se termina de desarrollar las demás funcionalidades consideradas para la solución al requerimiento.

- **PRUEBAS**

Las pruebas con el personal del área usuaria duran uno o dos días a lo sumo. Para realizar esto, lo primero que se realiza es la planificación de la logística de las pruebas. La gran mayoría de ocasiones el usuario es quien debe trasladarse a las oficinas de Quito para realizar las pruebas, pero dependiendo del impacto del requerimiento el Administrador tiene que dirigirse al sector del país, si fuese el caso, u organizar una reunión para llevar a cabo las pruebas.

Los usuarios son quienes tienen los datos en papeles manuales, los mismos que le sirven al administrador para ingresarlos en el dar inicio a las pruebas sobre la solución desarrollada. Entonces, el administrador ingresa de a poco la información y prueba cada proceso (Ej: se ingresa un registro que después se puede visualizar, modificar o eliminar). Si se detecta una falencia en el código fuente, o se requiere modificar el diseño para que se visualice la información de forma distinta, entonces este cambio es realizado simultáneamente a su identificación y se continúa con las pruebas. Al finalizar, se hacen actas acerca del término de las pruebas totales y se comunica con las áreas usuarias, administrativas u operativas para verificar que los cambios han sido exitosos.

- **IMPLEMENTACION, EVALUACION Y MANTENIMIENTO**

Una vez realizada la implementación del requerimiento, se procede con la capacitación del personal de las distintas áreas usuarias. Inclusive, y si el requerimiento así lo específico en sus primeras etapas, la capacitación se realiza a nivel nacional. Para esto el administrador de la aplicación se encarga de coordinar con los distintos jefes departamentales, la logística para realizar dicha capacitación. El tiempo estimado para realizar esto varía siempre, y es política de la empresa el prestar mayor atención a una capacitación del personal clave de cada área, para que ellos reproduzcan el conocimiento a las personas que desempeñan funciones operarias propiamente.

Es importante saber que la capacitación inicia desde la fase de pruebas, pero de forma implícita. Dependiendo del impacto del requerimiento la capacitación puede darse vía telefónica, ya que el menú de opciones de la aplicación es el mismo para cada módulo y la gente ya está adiestrada con el uso del mismo.

2. ROLES, CARGOS Y PERSONAS

Del relevamiento realizado se nos informó que las personas que intervienen en el proceso de desarrollo son:

Nombre	Cargo	Rol	Fase
Ing. Julio Delgado	Especialista Informático IIIB	Jefe del Proyecto	Todas
		Analista de Sistemas	Análisis, Diseño
		Arquitecto de Software	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador	Desarrollo
		Integrador	Desarrollo, Pruebas
		Implementador	Implementación
Ing. Fernando Grijalva	Especialista Informático IIIB	Jefe de Proyecto	Todas
		Analista de Sistemas	Análisis
		Desarrollador	Desarrollo
Ing. Blanca Rivera	Especialista Informático IIIB	Jefe de Desarrollo	-------
		Jefe de Proyecto	Todas
		Analista de Sistemas	Análisis
		Desarrollador	Desarrollo
Usuario	Personal del Área Usuaria	Verificador (persona que realizar las pruebas)	Pruebas

Tabla 3.9. Descripción de los roles, cargos y personas que intervienen en el desarrollo de software con COBOL y COBOL Cics
Realizado por: Xavier Salazar / Eduardo Velalcázar

3.1.2.1. DIAGRAMAS DE FLUJO DEL PROCESO DE DESARROLLO

a) Proceso General de Desarrollo de Software

- **Diagrama de Flujo**

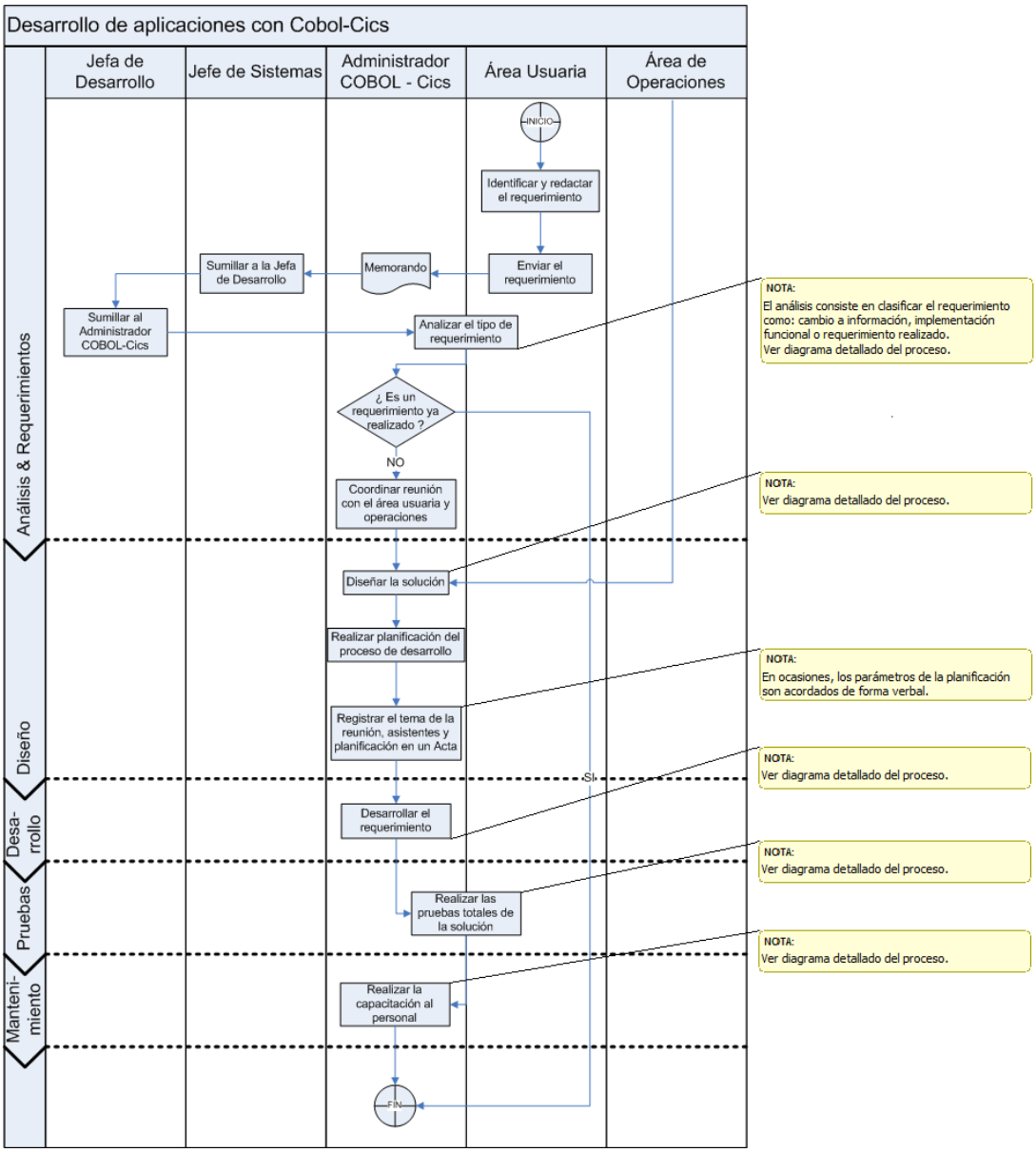
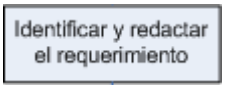
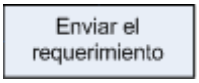
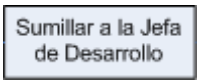
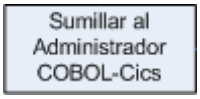
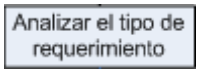
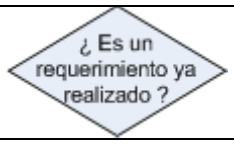
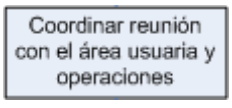


Gráfico 3.7. COBOL Cics: Proceso General de Desarrollo de Software
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de fases, elementos y responsables**

Elemento	Descripción	Responsable	Fase
	El requerimiento es identificado a raíz de una necesidad y los jefes de área se encargan de dar forma a la solicitud inicial realizan un análisis rápido y generalizado.	Jefe de Área Usuaria	Requerimientos
	El requerimiento debe ser entregado por el órgano regular de la compañía; es decir, la comunicación a nivel departamental.	Jefe de Área Usuaria	Requerimientos
	Documento en el que se redacta el requerimiento y que incluye una descripción general con los rasgos más importantes del mismo.	Jefe de Área Usuaria	Requerimientos
	Acción en la que el Jefe de Sistemas realiza la asignación de responsabilidades al área y personal pertinente, dentro de la Unidad de Sistemas.	Jefe de Sistemas	Requerimientos
	Acción en la que la Jefa de Desarrollo se encarga de asignar las responsabilidades al Administrador de la aplicación.	Jefa de Desarrollo	Requerimientos
	El análisis consiste en realizar una clasificación acerca del tipo de requerimiento. El resultado final del análisis define si el requerimiento es: un cambio que se necesita hacer a la información, una implementación funcional o una solicitud realizada e implementada con alguna anterioridad.	Administrador COBOL-Cics	Requerimientos
	Decisión tomada de acuerdo al análisis realizado por el Administrador.	Administrador COBOL-Cics	Requerimientos
	La reunión tiene como objetivo identificar y convocar a todos los actores del proceso para	Administrador COBOL-Cics	Requerimientos

	realizar el refinamiento del requerimiento.		
Diseñar la solución	El diseño es realizado por el Administrador pero es el resumen de la opinión general de todos los involucrados.	Administrador COBOL-Cics	Diseño
Realizar planificación del proceso de desarrollo	La planificación especifica el límite de tiempo para la entrega del producto de software.	Administrador COBOL-Cics	Diseño
Registrar el tema de la reunión, asistentes y planificación en un Acta	Los datos registrados son: nombre y cargo de los que intervienen, tema de la reunión y, generalmente, las conclusiones sobre la planificación.	Administrador COBOL-Cics	Diseño
Desarrollar el requerimiento	Dar inicio al proceso de desarrollo del producto o solución de software.	Administrador COBOL-Cics	Desarrollo
Realizar las pruebas totales de la solución	Someter los procesos desarrollados a pruebas que avalen la eficacia de los mismos, con mediciones sobre la consistencia de la información durante todo el proceso.	Administrador COBOL-Cics / Personal del Área Usuaria	Pruebas
Realizar la capacitación al personal	La capacitación se realiza a los usuarios finales o Jefes de Área de los Departamentos involucrados para que se transmita el conocimiento.	Administrador COBOL-Cics	Mantenimiento

Tabla 3.10. COBOL_Cics: Proceso General de Desarrollo de Software Descripción de fases, elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

b) Diagramas detallados en cada fase

Fase: ANÁLISIS Y REQUERIMIENTOS

Proceso: Analizar el tipo de requerimiento

- **Diagrama de Flujo**

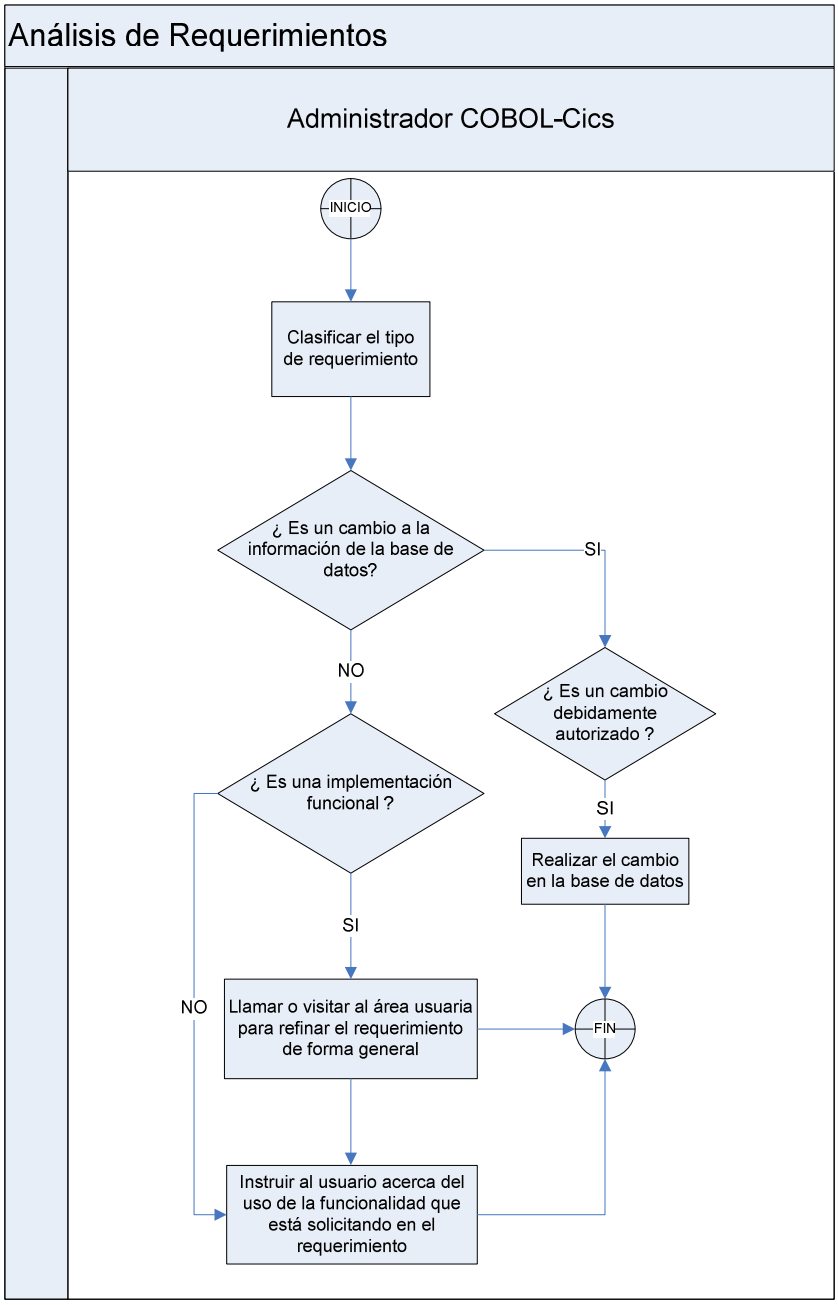


Gráfico 3.8.COBOL Cics: Proceso Análisis de Requerimientos
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables del proceso**

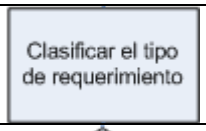
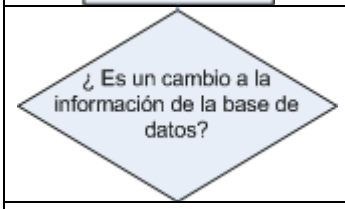
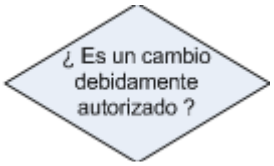
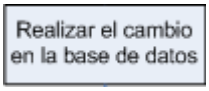
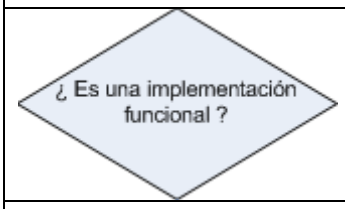
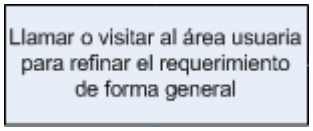
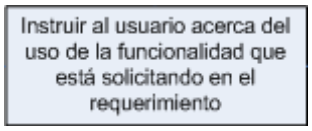
Elemento	Descripción	Responsable
	Acción que consiste en identificar el tipo de requerimiento que se está solicitando.	Administrador COBOL-Cics
	Decisión tomada de acuerdo a la clasificación que se realiza sobre el tipo de requerimiento.	Administrador COBOL-Cics
	Los cambios a la base de datos que se solicitan por memorándum son de impacto importante, por ejemplo: el reingreso de un registro a otro código de un producto. Sin embargo, este tipo de requerimiento no se presenta comúnmente, y la decisión es tomada de acuerdo al impacto del cambio y a la evidencia presentada (papeles físicos).	Administrador COBOL-Cics
	El Administrador ingresa a la base de datos de la aplicación y procede con la realización del cambio solicitado.	Administrador COBOL-Cics
	Decisión tomada de acuerdo al análisis realizado acerca del tipo de requerimiento.	Administrador COBOL-Cics
	El Administrador de la aplicación realiza un relevamiento general con el fin de refinar el requerimiento con la definición de parámetros propios de desarrollo (identificación del módulo que se afectará).	Administrador COBOL-Cics
	Si el requerimiento solicitado ya está implementado pero el usuario lo desconoce, el administrador realiza una capacitación breve por teléfono acerca de generalidades, ubicación, etc. de la funcionalidad.	Administrador COBOL-Cics

Tabla 3.11. COBOL Cics: Proceso Análisis de Requerimientos
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fase: DISEÑO

Proceso: Diseñar la solución

- **Diagrama de Flujo**

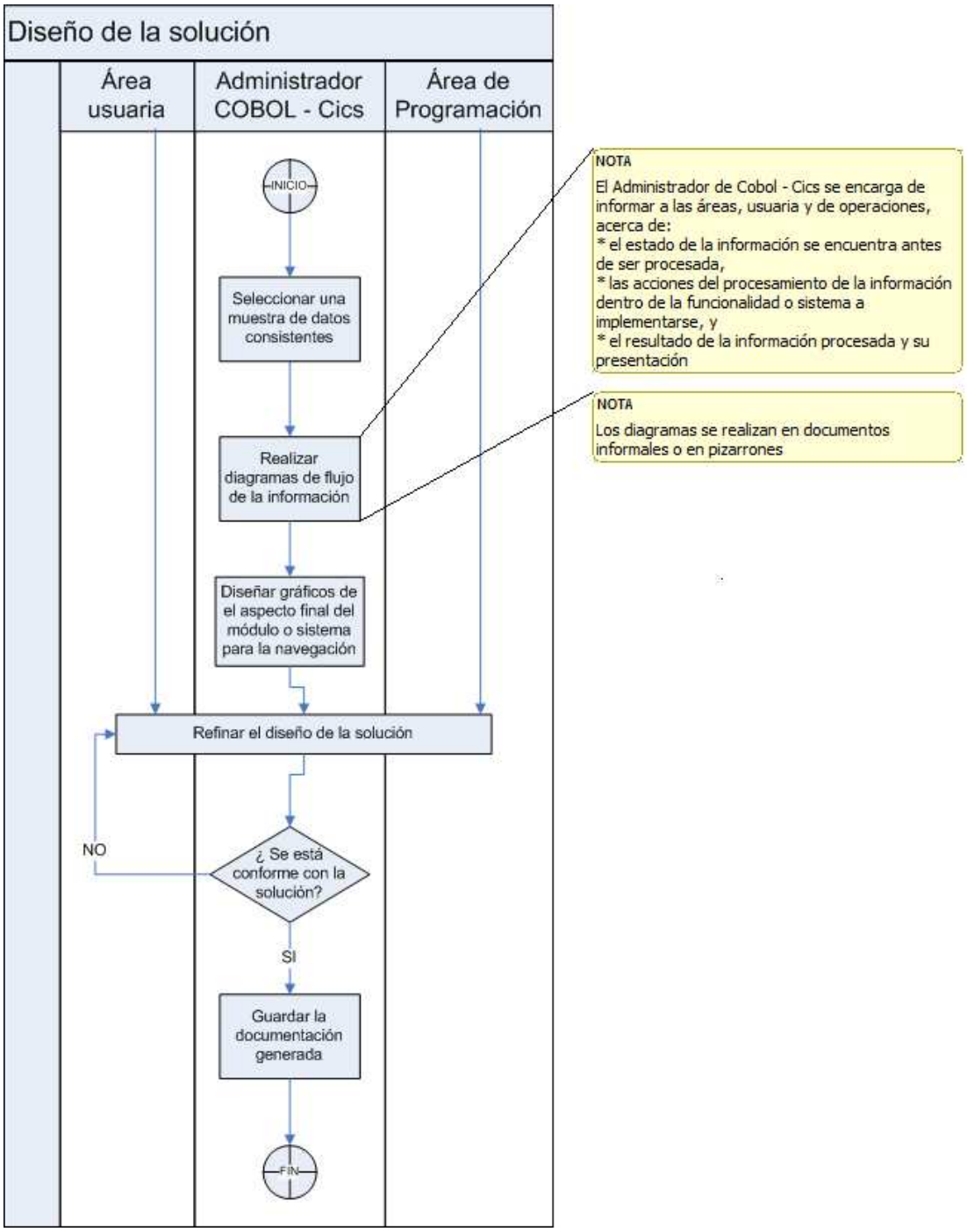


Gráfico 3.9. COBOL Cics: Proceso Diseñar la Solución
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

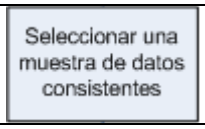
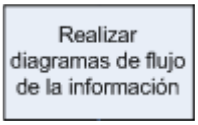
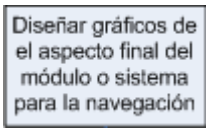
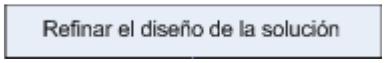
Elemento	Descripción	Responsable
	La selección de datos debe ser prolija para que el diseño refleje la realidad lo mejor posible.	Administrador COBOL-Cics
	Con los datos seleccionados se realizan flujos de procesos y de actividades en el pizarrón, o en documentos informales, para visualizar todas las acciones del proceso a desarrollarse (en el caso que se deba desarrollar un módulo nuevo, esta acción dura más tiempo).	Administrador COBOL-Cics
	El administrador realiza un diseño de la interfaz de navegación, como un mapa, para que los usuarios sepan como acceder a todas las funcionalidades del módulo.	Administrador COBOL-Cics
	Como producto del diseño de la solución, se realiza el refinamiento del requerimiento inicial y se le da cotas y alcance a la solución a desarrollar.	Administrador COBOL-Cics
	Decisión tomada de acuerdo al diseño realizado.	Administrador COBOL-Cics
	La documentación que se genera del diseño es guardada por el Administrador de la aplicación puesto que le sirve de referencia para realizar el desarrollo de la solución.	Administrador COBOL-Cics

Tabla 3.12. COBOL Cics: Proceso Diseñar la Solución
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fase: DESARROLLO

Proceso: Desarrollar la solución

- **Diagrama de Flujo**

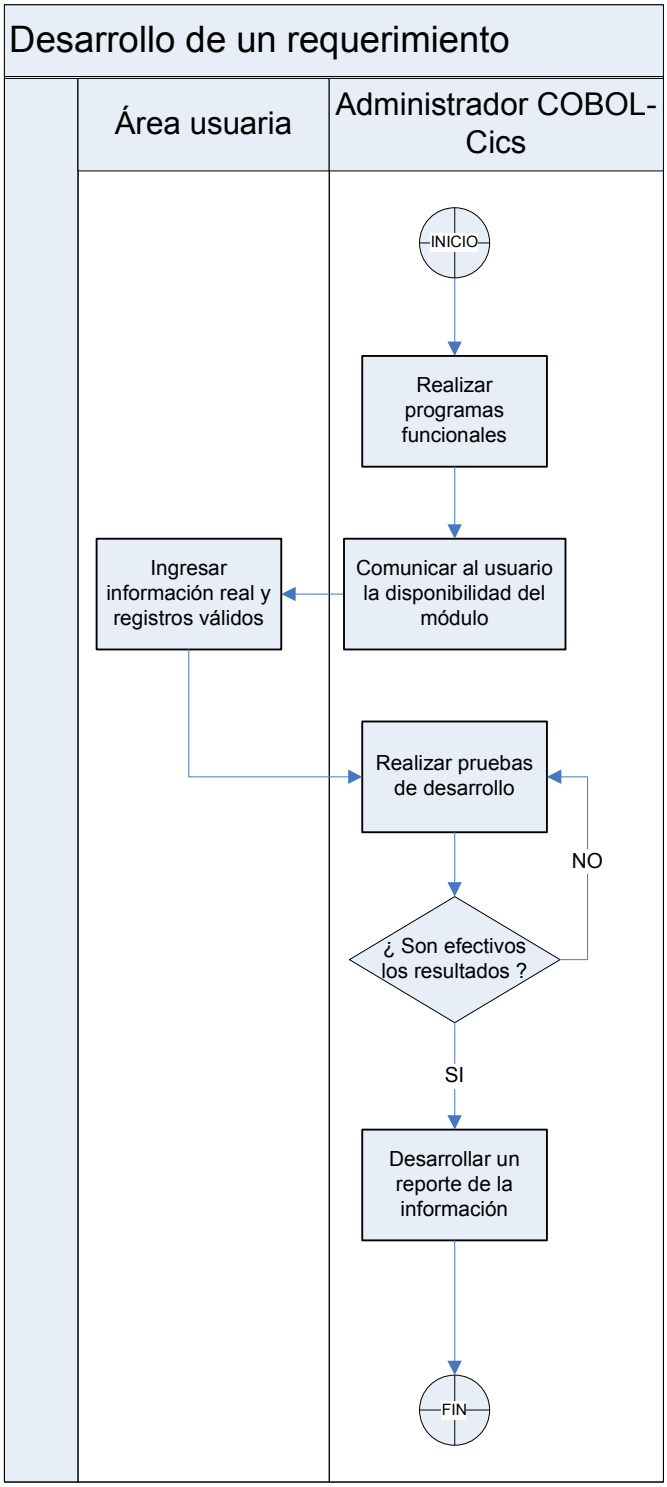


Gráfico 3.10. COBOL Cics: Proceso Desarrollo de la Solución
Diagrama de flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

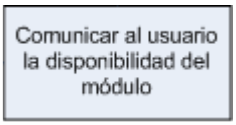
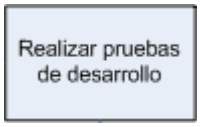
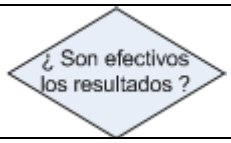
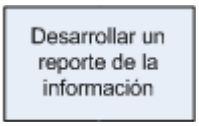
Elemento	Descripción	Responsable
	Los programas funcionales son códigos de programación en los que se empieza a solucionar el requerimiento. Igualmente se crean pantallas en las que se ingresa información para realizar las pruebas modulares.	Administrador COBOL-Cics
	Una vez desarrollado el módulo con la funcionalidad solicitada, se comunica al usuario la disponibilidad del requerimiento dentro del sistema para que éste proceda con el ingreso de datos que servirán para la realización de pruebas y correcciones en el desarrollo.	Administrador COBOL-Cics
	El personal del área usuaria ingresa información que reside en papeles físicos de forma masiva; así el Administrador podrá realizar pruebas con tiempos reales de respuesta al procesamiento de sus módulos.	Área Usuaria
	El Administrador, entonces, realiza pruebas que consisten en validar la consistencia de los datos y la lógica de los procesos mediante un análisis del impacto en las tablas afectadas.	Administrador COBOL-Cics
	Decisión tomada de acuerdo a la efectividad de las pruebas realizadas.	Administrador COBOL-Cics
	Para que quede constancia del desarrollo de la funcionalidad y las pruebas realizadas, el Administrador registra estos hechos en documentos propios que son resguardados en su poder.	Administrador COBOL-Cics

Tabla 3.13. COBOL Cics: Proceso Desarrollo de la Solución
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fase: PRUEBAS

Proceso: Realizar las pruebas totales de la solución

- **Diagrama de Flujo**

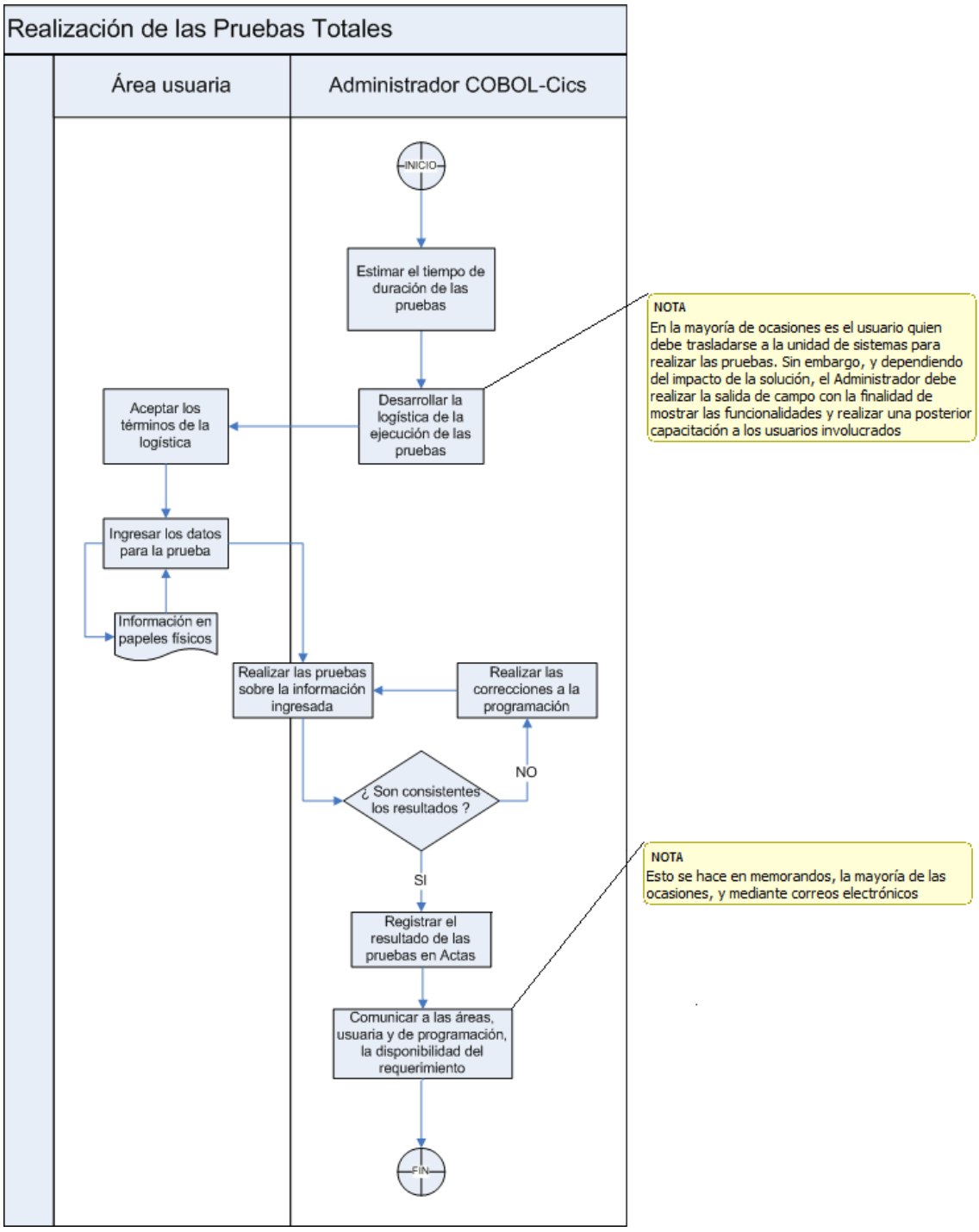
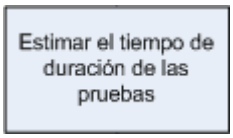
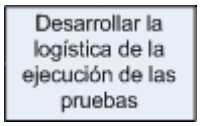
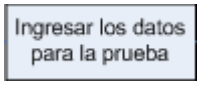
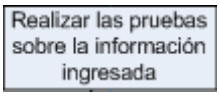


Gráfico 3.11. COBOL Cics: Proceso Realizar las Pruebas Totales
Diagrama de flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

Elemento	Descripción	Responsable
	El Administrador estima el tiempo que durarán las pruebas de acuerdo a su experiencia. Sin embargo, ninguna prueba dura más allá de dos días	Administrador COBOL-Cics
	La logística hace mención, principalmente, al traslado del personal para realizar las pruebas. En la mayoría de ocasiones, el usuario se traslada a la Unidad de Sistemas para realizar las pruebas. Sin embargo, y dependiendo del impacto de la solución, el Administrador debe realizar la salida de campo con la finalidad de mostrar las funcionalidades y capacitar posteriormente a los usuarios involucrados	Administrador COBOL-Cics
	Usualmente el usuario acepta la logística del traslado. Las excepciones se suscitarían en el caso que la carga de trabajo impida a una de las partes trasladarse; en ese caso se acuerda una fecha oportuna	Área Usuaria
	Una vez que la logística se lleva a cabo, y ya en la prueba, el personal del área usuaria ingresa datos para dar inicio a la manipulación de los mismos	Área Usuaria
	Los papeles físicos son documentos que contienen información de los registros que serán procesados en los módulos desarrollados	_____
	Las pruebas consisten en la manipulación de los datos que fueron ingresados. Esta acción es ejecutada con el usuario, ya sea de forma presencial o a vía telefónica, pero es un proceso que mide el resultado del procesamiento de la información	Administrador COBOL-Cics / Área Usuaria

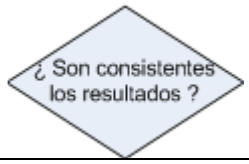
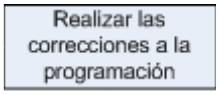
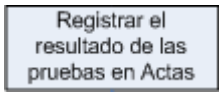
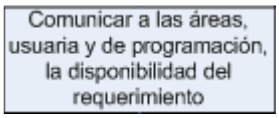
	Decisión tomada de acuerdo a los resultados obtenidos de las pruebas realizadas	Administrador COBOL-Cics
	En caso de que los resultados no satisfagan a alguna de las dos partes, se procede a realizar las correcciones en el código de programación en busca de resultados efectivos en las pruebas	Administrador COBOL-Cics
	Una vez terminado el proceso de pruebas, el Administrador realiza un acta en la que se registra la finalización de dichas pruebas y la entrega del producto terminado de software	Administrador COBOL-Cics
	Usualmente el acta redactada es de conocimiento de las áreas involucradas; sin embargo, se realizan comunicados vía correo electrónico, telefónica, memos, etc. para asegurarse de que la implementación sea de conocimiento general	Administrador COBOL-Cics

Tabla 3.14. COBOL Cics: Proceso Realizar las Pruebas Totales
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fase: MANTENIMIENTO

Proceso: Realizar la capacitación al personal

- **Diagrama de Flujo**

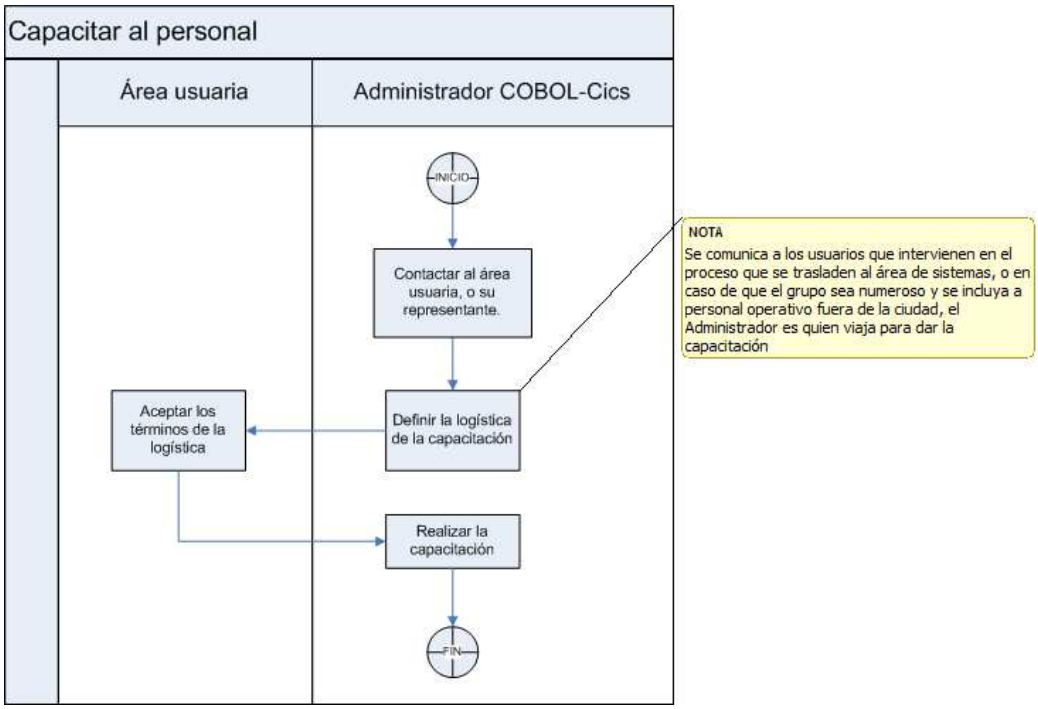


Gráfico 3.12. COBOL Cics: Proceso Capacitar al Personal
Diagrama de flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

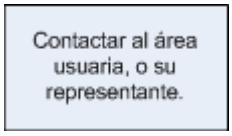
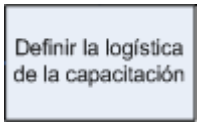
Elemento	Descripción	Responsable
	El Administrador acuerda con el Jefe de Área en realizar una capacitación sobre el nuevo sistema.	Administrador COBOL-Cics
	Se define el traslado, tiempo que se invertirá en la capacitación y personal que accederá a la misma.	Administrador COBOL-Cics
	El representante del área usuaria acepta los términos de la logística de acuerdo a su carga de trabajo.	Área Usuaria
	El Administrador se encarga de desarrollar y ejecutar la capacitación al personal.	Administrador COBOL-Cics

Tabla 3.15. COBOL Cics: Proceso Capacitar al Personal
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

3.1.3. LOTUS

HISTORIA Y DESCRIPCIÓN GENERAL DEL LENGUAJE

IBM Lotus Notes es una herramienta desarrollada por la casa IBM para la administración de funciones de escritorio, además de ser el principal cliente integrado para IBM Lotus Domino, que combina funciones acerca del correo electrónico, calendario y planificación, con una potente plataforma de escritorio para aplicaciones de colaboración.

El software Lotus Notes ofrece importantes herramientas de colaboración empresarial diseñadas para aumentar la productividad individual y en equipo, así como la capacidad de respuesta de la empresa en general.

Entre las principales ventajas de esta herramienta, tenemos que LOTUS NOTES:

- Maximiza la capacidad de respuesta de los empleados ya que pone a su disposición un único punto de acceso a los recursos de uso habitual tales como el correo electrónico, el calendario, la mensajería instantánea y las herramientas de colaboración.
- Facilita la toma de decisiones mediante colaboración al incorporar la notificación de presencia, la mensajería instantánea y las conferencias Web (opcional).
- Presenta funciones sólidas de seguridad que minimizan o eliminan el efecto de los virus informáticos. Incluye funciones de productividad, como la búsqueda texto completo, diseñadas para que los usuarios gestionen el correo de forma más eficaz. Ofrece excelentes funciones

de planificación y calendario, tales como la gestión centralizada de salas de conferencias y recursos para reuniones como, por ejemplo, equipo audiovisual.

- Incorpora tecnología avanzada de replicación que aumenta la productividad y la eficacia de los usuarios, incluso cuando no están conectados a la red.

- **ESTRUCTURA Y CONCEPTUALIZACIÓN DE LOS SISTEMAS DESARROLLADOS EN LOTUS NOTES**

Los sistemas desarrollados en LOTUS NOTES son:

- *SISTEMA DE CONTROL DE ASISTENCIA*
- *SISTEMAS DOCUMENTALES: Administración documental, Órdenes de Pago, Viáticos, Garantías y Seguros, Secretaría General*

Todos los sistemas tienen características similares por lo que haremos énfasis en el sistema documental de órdenes de pago y el sistema de viáticos.

El sistema de Órdenes de Pago es un sistema que se encarga de administrar los documentos corporativos referentes a las órdenes de pago por todos los conceptos mediante la generación de un flujo centralizado de procesamiento en el que se considera una estructura ordenada que contiene información acerca del Departamento en el que se está procesando, autorizando o resguardando un documento. Esto permite que los usuarios puedan conocer el estado de procesamiento de sus documentos con especificaciones de las

fechas en las que se traslada de un departamento a otro, o la fecha de aprobación del mismo.

- **ELEMENTOS QUE INTERVIENEN EN EL PROCESO DE DESARROLLO DE SISTEMAS EN LOTUS NOTES**

1. **FASES**

    - **ANÁLISIS**

En esta fase, el personal del área usuaria detecta una necesidad y genera un requerimiento. La forma de reportar el requerimiento depende del impacto del mismo; es decir, si el cambio es considerado menor, entonces el usuario reporta el requerimiento vía telefónica.

Estos cambios solicitados, pueden clasificarse comúnmente en:

- Errores de digitación
- Problemas en la ejecución de los procesamientos de lotes de información o procesos batch, por errores en el servidor AS/400, para los distintos sistemas

En el caso que el procesamiento de la información no ha sido exitoso, las personas del área usuaria reportan el error telefónicamente y se procede a realizar el cambio inmediatamente.

Durante el relevamiento de información realizado por la analista de sistemas se genera un documento denominado Visión, el cual fue creado con la intención de documentar las características más importantes del requerimiento en una plantilla que indaga acerca de la base de datos, las técnicas para la captura de requisitos, ya sea por casos de uso o diagramas

de proceso. Al final, el usuario debe realizar la aprobación de la Visión mediante una sumilla y a partir de allí se procede a desarrollar lo solicitado.

En esta misma etapa, además, se realiza un análisis en el que la Administradora identifica si el requerimiento solicitado es una función ya implementada, y si es el caso, entonces responde vía memorándum informando la situación para luego proceder con la capacitación pertinente.

En caso de que el requerimiento sea nuevo, la Administradora se dirige al área usuaria y mantiene una reunión con el solicitante con el fin de refinar y entender mejor el requerimiento. El Jefe de Sistemas conoce del requerimiento, y de acuerdo al impacto del mismo, designa a una persona (pasante) para que acompañe a la Administradora a realizar el relevamiento y posterior desarrollo de la solución al requerimiento.

Una de las partes importantes del análisis consiste en estudiar el requerimiento para decidir si es concerniente y del alcance de Lotus. Usualmente el resultado es afirmativo, pero siempre va a depender de la precisión con la que se realiza el relevamiento.

- **DISEÑO**

En la etapa de diseño, el primer paso es realizar un cronograma de actividades para organizar el posterior desarrollo. Este cronograma es realizado comúnmente en una de las herramientas informáticas: Microsoft Project. Sin embargo, el conocimiento del mismo es del equipo de desarrollo

de Sistemas y de la jefatura de área y departamento, mas no del área usuaria.

Otro aspecto de esta etapa es la utilización de un diagramador de clases, para visualizar la constitución que tendrá la estructura de base de datos. La herramienta utilizada, si el requerimiento así lo necesita, es Erwin. Esta herramienta permite realizar modelos entidad relación bastante completos; sin embargo, la utilización de la misma no está especificada en ninguna política por lo que la modelación de la estructura que tendrá la base de datos, es realizada informalmente en hojas a parte o simplemente concebidas por la Administradora.

En el diseño también se considera la clasificación de las distintas partes del requerimiento; esto quiere decir que se da prioridad a lo urgente y no a lo importante. Además, y considerando el análisis y la información en el documento Visión, se estudia el alcance del requerimiento para viabilizar la posibilidad de realizar casos de uso o diagramas de proceso, o si la tecnología que se piensa utilizar requerirá de herramientas para implementaciones en la WEB.

Todo este diseño es añadido a las especificaciones de la Visión, para que el área usuaria apruebe, y consecuentemente de inicio la etapa de desarrollo de la solución al requerimiento.

- **DESARROLLO**

La asignación de responsabilidades la realiza la Administradora, quien clasifica las partes del desarrollo y decide los módulos que estarán a su cargo y los que serán responsabilidad del personal que brinda ayuda (pasante). Debido a la relación del pasante con la compañía, nunca se le asigna módulos que tomen mucho tiempo o conocimientos específicos.

Luego se realiza un cronograma de cambios, que es realizado por la persona que desarrolla ya que calcula el tiempo que le tomaría implementar la solución. Este cronograma es realizado empíricamente. Después empieza la generación del código y las pruebas modulares de desarrollo que son realizadas conjuntamente con el área usuaria. Según se nos informó, el usuario interviene en el proceso de desarrollo ya que comúnmente realiza modificaciones, no sustanciales, al requerimiento inicial.

Es decir, el aumento de un campo, o la modificación de la posición para su visualización. La etapa de desarrollo termina con la instalación de un programa piloto disponible para todos los usuarios.

- **PRUEBAS**

La etapa de pruebas no es considerada dentro del cronograma realizado en el desarrollo. Se tiene un ambiente de pruebas en el que utilizan tablas para realizar pruebas sobre el desarrollo.

En esta fase, es común encontrar que el usuario solicite la implementación de cambios o modificaciones al requerimiento inicial, y fuera de los ya

encontrados en la fase anterior de pruebas modulares de desarrollo, por lo que se procede a realizar los cambios en ese instante. El monitoreo del éxito sobre las pruebas es realizado telefónicamente, donde el usuario podría notificar la aparición de algunas oportunidades de mejora al diseño o en sí anomalías en la solución desarrollada. De ser el caso, se realiza la depuración del desarrollo o el desarrollo de la oportunidad de mejora.

Suponiendo que la prueba sea eficaz, se escribe un memorándum dirigo del departamento de sistemas al área usuaria, donde se explica que el usuario ha aprobado y aceptado la culminación de las pruebas. Además se notifica la disponibilidad del requerimiento en el ambiente de producción.

- **IMPLEMENTACIÓN, EVALUACIÓN Y MANTENIMIENTO**

Si la implantación se debe realizar para un área en específico, se proporciona una capacitación general al personal del área usuaria entregando manuales de usuario (requerimiento de la Contraloría General del Estado). Sin embargo, y en caso de que la implantación sea para la filial, la implantación podría tomar mucho tiempo, ya que la Administradora deberá dirigirse a las distintas zonas del país con la finalidad de realizar la instalación del mismo, probar la funcionalidad de la solución midiendo tiempos de respuesta, y entrenando al personal clave de las distintas áreas en cómo utilizar el nuevo requerimiento implantado.

La filial está dividida en dos grandes zonas, la norte y sur, por lo que si dicha instalación debe realizarse en una dependencia de la zona sur, entonces la

Administradora realizará la capacitación y entregará las bases de datos correspondientes al personal pertinente del Departamento de Sistemas de la zona sur, quienes a su vez darán soporte y realizarán el mantenimiento del sistema.

Luego, se elabora un acta de entrega-recepción que es firmado por el responsable del área que solicitó el requerimiento.

Como trabajo adicional, y con el apoyo del personal clave del área usuaria, se realizan indexaciones para que los motores de búsqueda de documentos optimicen sus recursos.

Si existen cambios luego de que el sistema ha sido puesto en producción, la Administradora clasifica los mismos y recomienda al área usuaria que generen un nuevo requerimiento; o si es el caso, realiza las correcciones en el sistema desplegado (Ej: añadir un campo a un registro, incluir un rango de fechas para búsquedas, etc.)

2. ROLES, CARGOS Y PERSONAS

Del relevamiento realizado se nos informó que las personas que intervienen en el proceso de desarrollo son:

Nombre	Cargo	Rol	Fase
Ing. Lorena Burgos	Especialista de Sistemas IIIA	Jefe del Proyecto	Todas
		Analista de Sistemas	Análisis, Diseño
		Arquitecto de Software	Análisis, Diseño
		Desarrollador	Desarrollo
		Integrador	Desarrollo, Pruebas
		Implementador	Implementación
Cristina Zabala	Pasante	Analista de Sistemas	Análisis
		Desarrollador	Desarrollo
Usuario	Personal del Área Usuaria	Verificador (persona que realizar las pruebas)	Pruebas

Tabla 3.16. Descripción de los roles, cargos y personas que intervienen en el proceso de desarrollo con LOTUS NOTES
Realizado por: Xavier Salazar / Eduardo Velalcázar

3.1.3.1. DIAGRAMAS DE FLUJO DEL PROCESO DE DESARROLLO

a) Proceso General de Desarrollo de Software

- **Diagrama de Flujo**

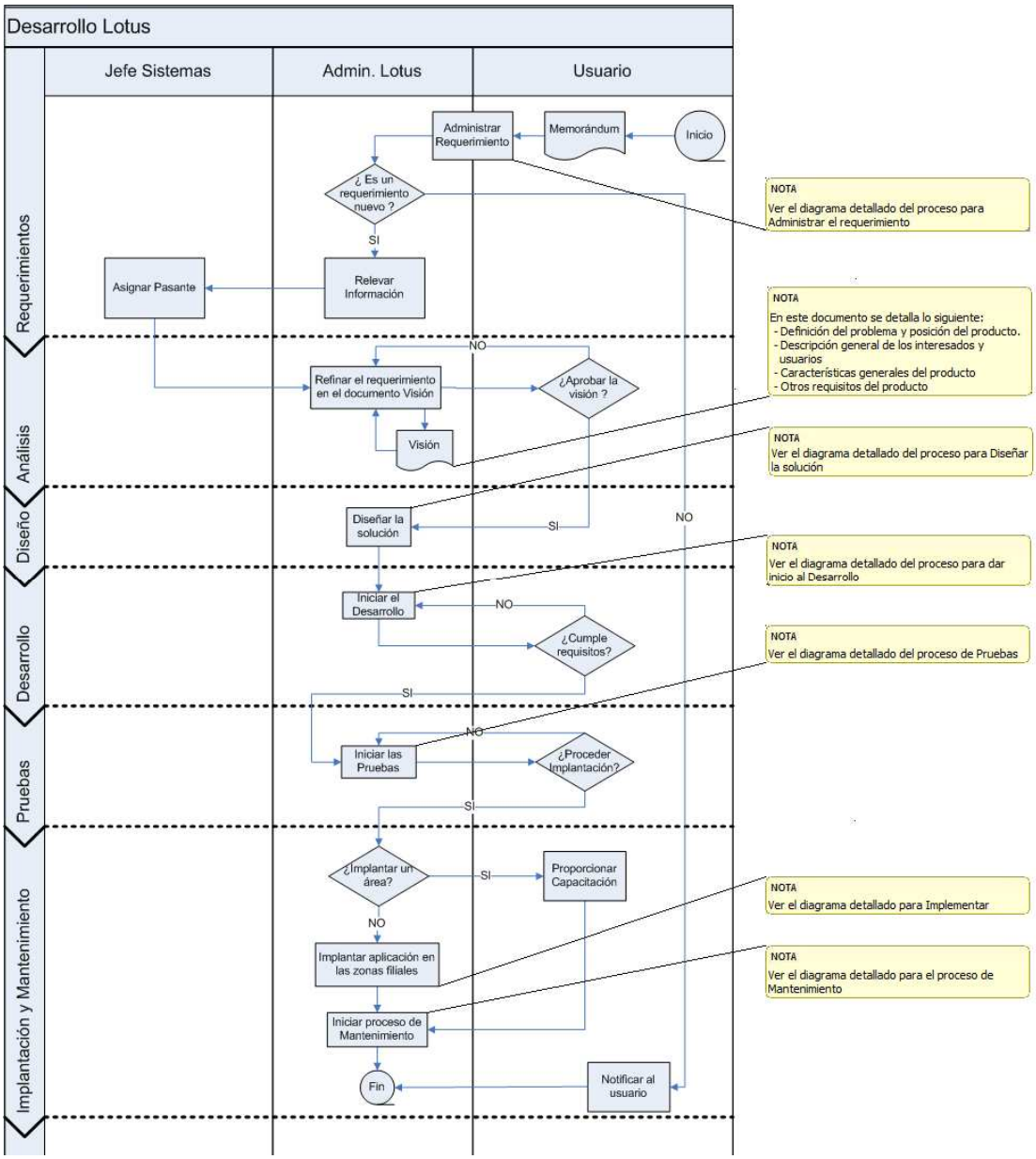
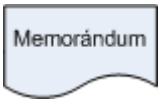
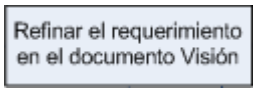
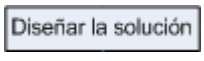
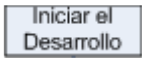


Gráfico 3.13. Lotus Notes: Proceso General de Desarrollo de Software
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de fases, elementos y responsables**

Elemento	Descripción	Responsable	Fase
	Documento en el que se redacta el requerimiento y que incluye una descripción general con los rasgos más importantes del mismo.	Jefe de Área Usuaria	Requerimientos
	Acción en la que se analiza el requerimiento por parte del Especialista para validar su disposición y necesidad.	Jefe de Área Usuaria / Administradora Lotus Notes	
	Decisión tomada de acuerdo al análisis realizado por el Administrador.	Administradora Lotus Notes	
	Conversaciones telefónicas o visitas para ampliar el espectro de comprensión del requerimiento.	Administradora Lotus Notes	
	La administradora escoge un pasante que conformará el equipo de desarrollo.	Administradora Lotus Notes	
	Se hace las especificaciones técnicas del requerimiento en el formato Visión.	Administradora Lotus Notes	Análisis
	Decisión tomada de acuerdo al análisis del documento Visión.	Jefe o personal de Área Usuaria	
	Documento que detalla, entre otras cosas, la definición del problema, características generales del producto, los interesados y los usuarios	--	
	El diseño es realizado por el Administrador pero es el resumen de la opinión general de todos los involucrados	Administradora Lotus Notes	Diseño
	Se utiliza los lenguajes Lotus script, Java o Java script. Las pruebas por cada módulo se realizan conjuntamente con el personal del área usuaria	Administradora Lotus Notes	Desarrollo
	Si el usuario se encuentra conforme con el avance entregado, prosigue la programación previamente planificada	Jefe o personal de Área Usuaria	

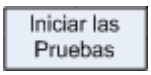
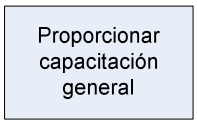
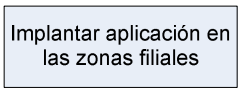
	Se instala una versión piloto en ambiente de pruebas para someter los procesos desarrollados a pruebas que avalen la eficacia de los mismos	Administradora Lotus Notes	Pruebas
	Decisión tomada por el área usuaria. El registro de las pruebas se realiza vía telefónica y se pueden dar oportunidades de mejora, ya sean al diseño o anomalías en el desarrollo de la aplicación. En caso de que el defecto sea encontrado se procederá a elaborar una depuración de cambios	Jefe o personal de Área Usuaria	Pruebas
	Decisión tomada luego de identificar si la implantación es sólo para un área en específico o es a nivel zonal	Administradora Lotus Notes	Implantación y Mantenimiento
	La capacitación se realiza a los usuarios finales o Jefes de Área de los Departamentos involucrados para que se transmita el conocimiento	Administradora Lotus Notes	Implantación y Mantenimiento
	La Administradora se desplaza a las distintas zonas e instala el software. Para descentralizar el soporte, se proporciona las bases necesarias al personal de Guayaquil y se elabora un acta de entrega-recepción, la cual es firmada por el responsable del área que solicitó el requerimiento	Administradora Lotus Notes	Implantación y Mantenimiento
	En caso de que se den nuevos requerimientos, se analiza su impacto y si es menor se implementa de forma inmediata; de ser significativo, se sigue el procedimiento expuesto	Administradora Lotus Notes	Implantación y Mantenimiento

Tabla 3.17. Lotus Notes: Proceso General de Desarrollo de Software Descripción de fases, elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

b) Diagramas detallados en cada fase

Fase: REQUERIMIENTOS

Proceso: Administrar el requerimiento

- **Diagrama de Flujo**

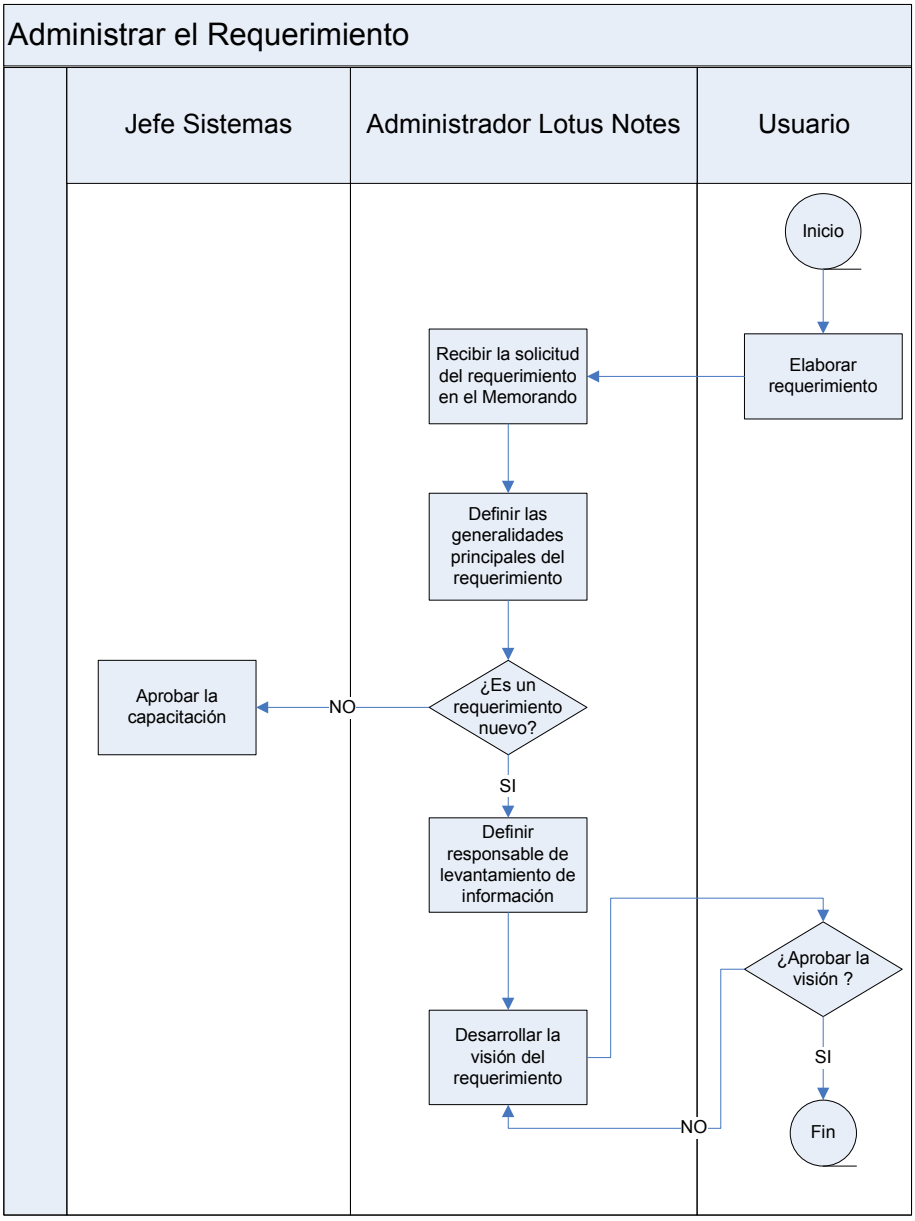


Gráfico 3.14. Lotus Notes: Proceso Administrar Requerimiento
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

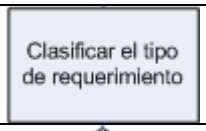
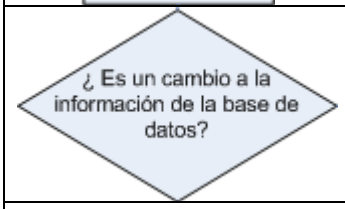
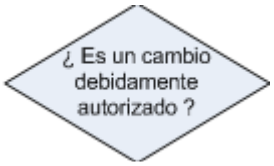
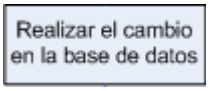
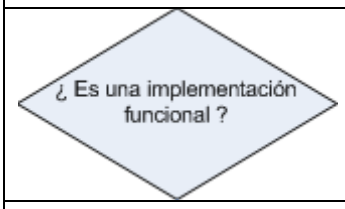
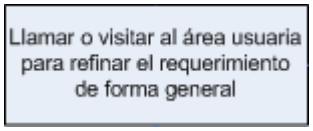
Elemento	Descripción	Responsable
	Acción que consiste en identificar el tipo de requerimiento que se está solicitando.	Administrador Lotus Notes
	Decisión tomada de acuerdo a la clasificación que se haga del tipo de requerimiento.	Administrador Lotus Notes
	Los cambios a la base de datos que se solicitan por memorándum son de impacto importante, por ejemplo: el reingreso de un registro a otro código de un producto. Sin embargo, este tipo de requerimiento no se presenta comúnmente, y la decisión es tomada de acuerdo al impacto del cambio y a la evidencia presentada (papeles físicos).	Administrador Lotus Notes
	El Administrador ingresa a la base de datos de la aplicación y procede con la realización del cambio solicitado.	Administrador Lotus Notes
	Decisión tomada de acuerdo al análisis realizado acerca del tipo de requerimiento.	Administrador Lotus Notes
	El Administrador de la aplicación realiza un relevamiento general con el fin de refinar el requerimiento con la definición de parámetros propios de desarrollo (identificación del módulo que se afectará).	Administrador Lotus Notes

Tabla 3.18. Lotus Notes: Proceso General de Desarrollo de Software Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fase: DISEÑO

Proceso: Diseñar la solución

- **Diagrama de Flujo**

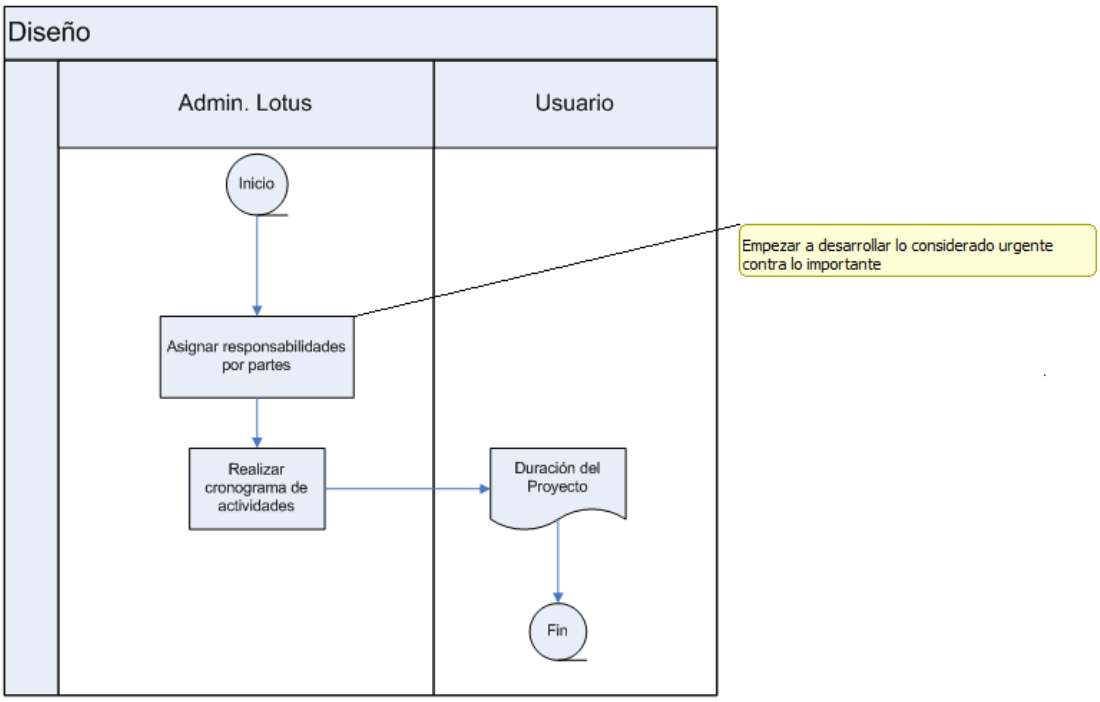


Gráfico 3.15. Lotus Notes: Proceso Diseñar la Solución
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de elementos y responsables**

Fase	Elemento	Descripción	Responsables
Diseño	Asignar responsabilidades por partes	Empezar a desarrollar lo considerado urgente contra lo importante	Administradora aplicación
	Realizar cronograma de actividades	Para realizar esto se utiliza la herramienta Microsoft Project.	Administradora aplicación
	Conocer el tiempo estimado duración proyecto	Se informa al usuario, el tiempo estimado de duración que tomará desarrollar la solución.	Área Usuaria

Tabla 3.19. Lotus Notes: Proceso Diseñar la Solución
Descripción de elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fases: **DESARROLLO** **PRUEBAS**

Procesos: Iniciar el desarrollo **Iniciar las pruebas**

- **Diagrama de Flujo**

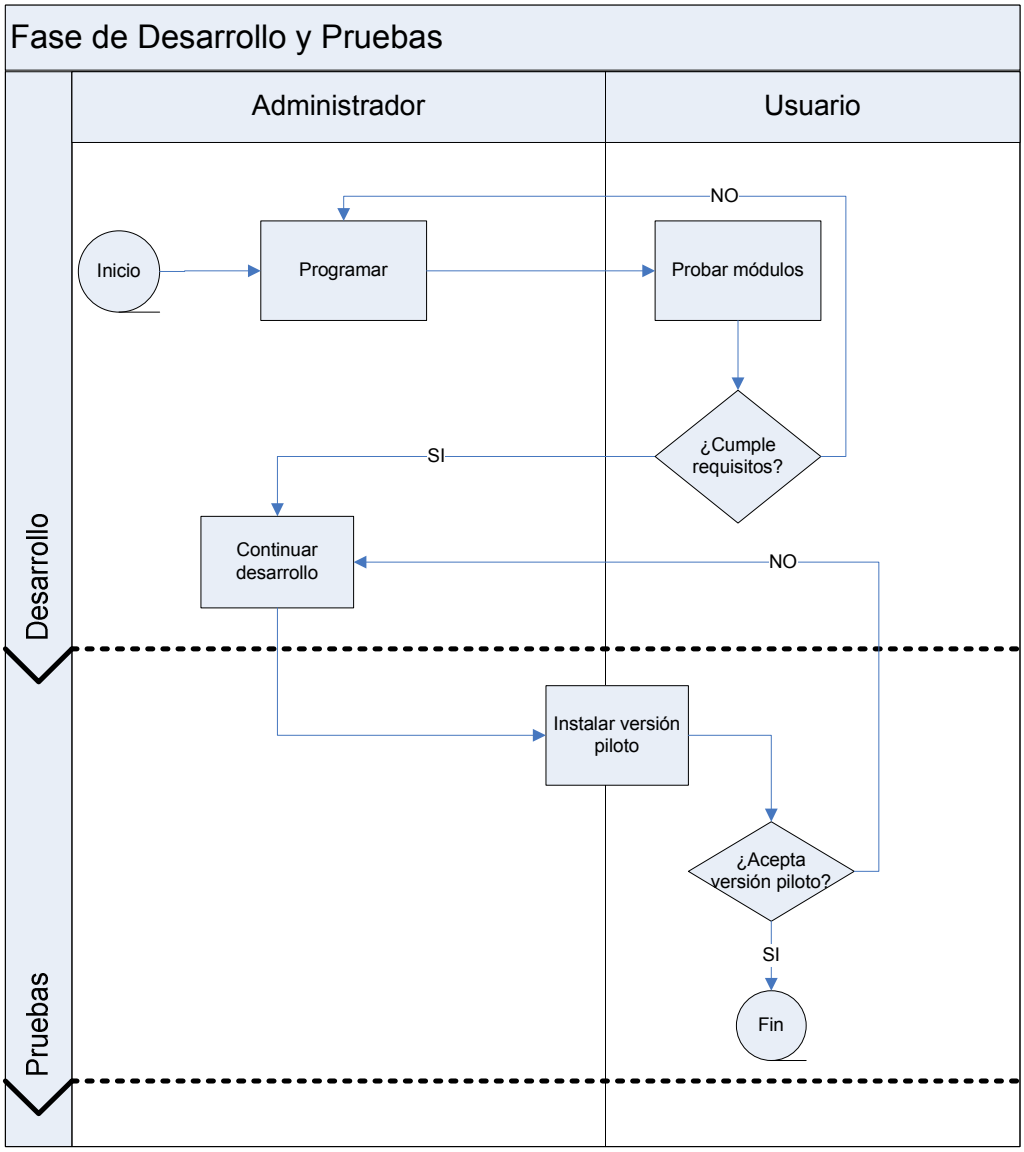


Gráfico 3.16. Lotus Notes: Procesos de Desarrollo y Pruebas
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de fases, elementos y responsables**

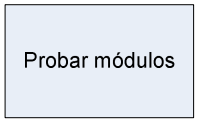
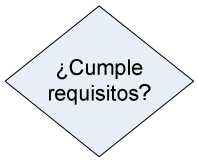
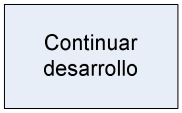
Elemento	Descripción	Responsables	Fase
	Se da inicio a la programación de la solución	Administradora LOTUS NOTES	Desarrollo
	Durante la etapa de desarrollo, las pruebas por cada módulo se realizan conjuntamente con el usuario	Administradora LOTUS NOTES / Área Usuaria	
	El usuario evalúa si el desarrollo cumple con los requisitos iniciales solicitados	Área Usuaria	
	Se ejecutan acciones que son parte de la planificación antes de la implementación de la versión piloto	Administradora LOTUS NOTES	
	Se instala la versión piloto en el ambiente de pruebas	Administradora LOTUS NOTES / Área Usuaria	Pruebas
	El usuario evalúa la versión antes de dejar la última versión instalada. El seguimiento a las pruebas se hace vía telefónica.	Área Usuaria	

Tabla 3.20. Lotus Notes: Procesos de Desarrollo y Pruebas
Descripción de fases, elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

Fases: **IMPLANTACIÓN** **MANTENIMIENTO**

Procesos: Implantar aplicación **Iniciar mantenimiento**

- **Diagrama de Flujo**

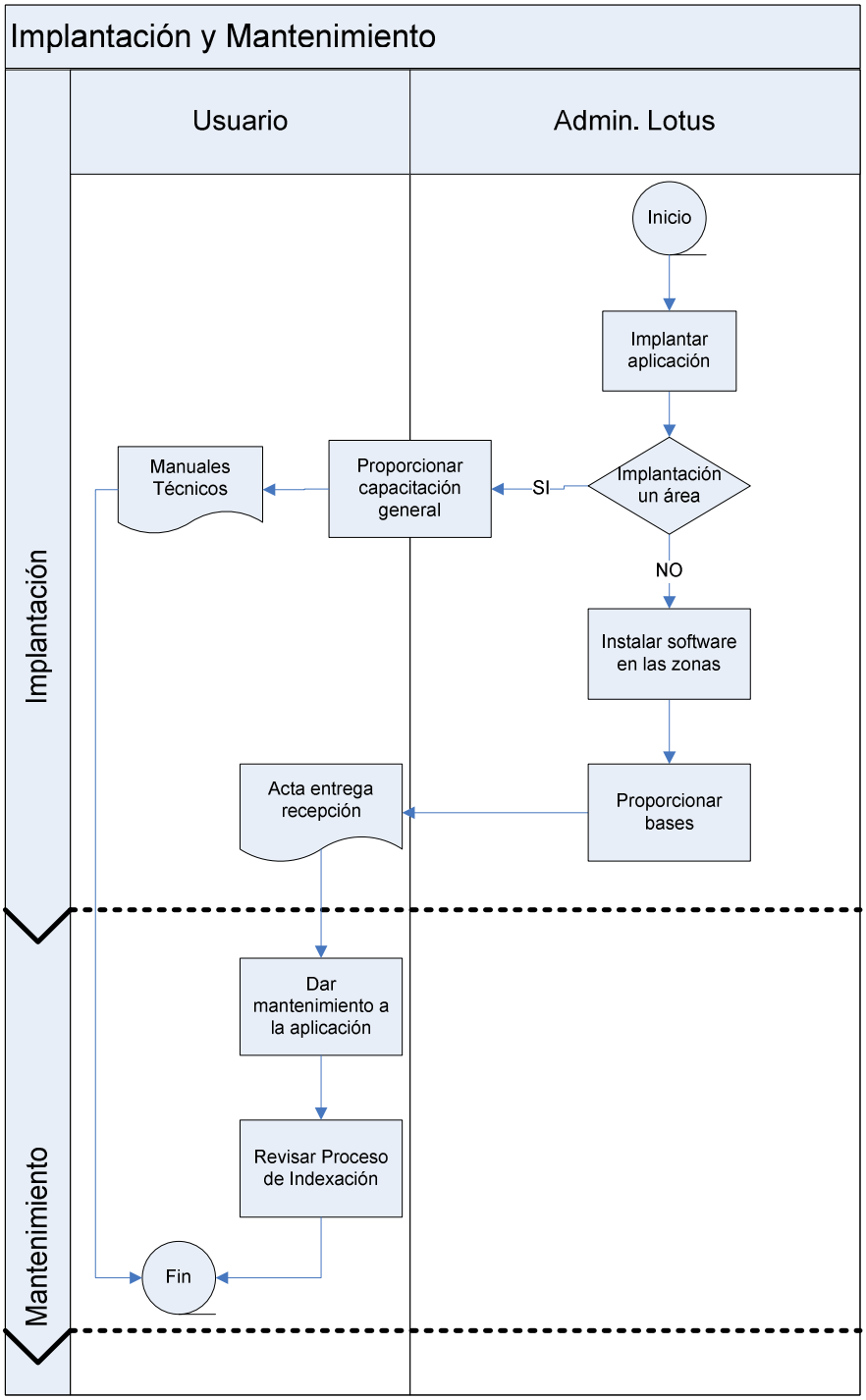
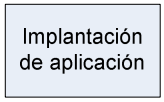
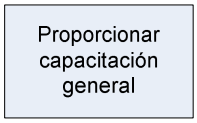
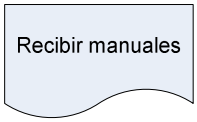
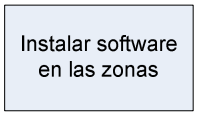
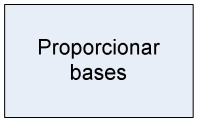
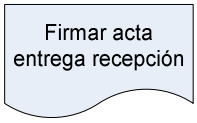
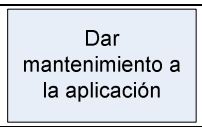


Gráfico 3.17. Lotus Notes: Procesos de Desarrollo y Pruebas
Diagrama de Flujo
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **Descripción de fases, elementos y responsables**

Elemento	Descripción	Responsable	Fase
	Se implementa la solución en ambiente de producción	Administradora LOTUS NOTES	Implantación
	Documento que valida la aprobación del usuario acerca del desarrollo de la solución	Administradora LOTUS NOTES	
	Se identifica si la implantación es solo para un área o a nivel zonal	Administradora LOTUS NOTES	
	Capacitación general al personal del área involucrada, definiendo la nueva aplicación desarrollada	Administradora LOTUS NOTES	
	Documentos acerca del manejo de las funciones de la nueva aplicación	Área Usuaria	
	La administradora se dirigirá a las distintas zonas para instalar la solución	Administrador LOTUS NOTES	
	Se proporcionará las bases para dar soporte y mantenimiento en una zona distinta de Quito	Administradora LOTUS NOTES	
	Se elabora un acta de entrega-recepción, la cual será firmada por el responsable del área que solicitó el requerimiento.	Área Usuaria	
	Los nuevos requerimientos que se susciten son	Administrador LOTUS NOTES	Mantenimiento

	tratados con el mismo ciclo de vida; de esto trata el mantenimiento de la aplicación. Si se tiene que corregir errores menores sobre desarrollos pasados, esto se realiza inmediatamente.		
Revisar proceso de indexación	Toda solución es sometida a una revisión de la indexación, ya que las bases son consultadas simultáneamente por un centenar de usuarios (en escenarios críticos)	Administrador LOTUS NOTES	

Tabla 3.21. Lotus Notes: Procesos de Implantar la Solución en las Filiales / Iniciar el Mantenimiento
Descripción de fases, elementos y responsables
Realizado por: Xavier Salazar / Eduardo Velalcázar

3.1.4. JAVA

- **HISTORIA Y DESCRIPCIÓN GENERAL DEL LENGUAJE**

Las características principales del lenguaje JAVA son las siguientes:

- Es un lenguaje orientado a objetos, por lo que diseña el software de tal forma que los distintos tipos de datos que usen, estén unidos a sus operaciones y sean accedidos únicamente mediante ellas.
- Independencia de la plataforma, significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware.
- El recolector de basura de Java permite una fácil creación y eliminación de objetos, mayor seguridad y puede que más rápida que lenguajes estructurados como C++.
- Java realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución.

Los programas no acceden directamente a la memoria del ordenador, siendo imposible que un programa escrito en Java pueda acceder a los recursos del ordenador sin que esta operación le sea permitida de forma explícita. De este modo, los datos del usuario quedan a salvo de la existencia de virus escritos en Java. La ejecución segura y controlada del código Java es una característica única, que no puede encontrarse en ninguna otra tecnología.

Es totalmente multiplataforma: Es un lenguaje sencillo, por lo que el entorno necesario para su ejecución es de pequeño tamaño y puede adaptarse incluso al interior de un navegador.

Las funciones de JAVA son aplicadas en los siguientes entornos:

- Dispositivos móviles y sistemas empotrados
- Navegador web
- Sistemas de servidor
- Aplicaciones gráficas de escritorio

- **ESTRUCTURA Y CONCEPTUALIZACIÓN DE LOS SISTEMAS DESARROLLADOS EN JAVA**

Los sistemas desarrollados en JAVA son:

- *SISTEMA COMERCIAL*
- *SISTEMA DE ABASTECEDORA (en proceso)*

El Sistema Comercial es el sistema que contiene los módulos relacionados a la administración de la contabilidad general y toda la lógica de las ventas, desde la compra de productos, inventarios para cada etapa de producción, ventas y pagos.

Si bien el sistema está estructurado para manejarse estáticamente en su forma general, el mismo considera la dinámica cuando se deben implementar regulaciones o parámetros que cambian constantemente (el precio del petróleo cambia de acuerdo a las ventas nacionales que se registran y a la demanda, por lo que hay que desarrollar módulos que administren esta lógica).

El Sistema Comercial fue diseñado en el lenguaje COBOL hace aproximadamente 30 años, y desde finales del 2006 se dio inicio al rediseño de ese sistema utilizando JAVA y otros lenguajes que permiten, eventualmente, migrar la aplicación a una plataforma basada en la WEB.

Actualmente el rediseño del Sistema Comercial, en sus partes funcionales principales, ya culminó, por lo que los desarrollos de nuevos módulos son prácticamente ninguno. Sin embargo el número de requerimientos que afinen la lógica que actualmente rige al Sistema Comercial (por ejemplo, se requiere de un módulo que administre los prepagos de los clientes y no permita realizar compras si las fechas de los prepagos han vencido) se ha incrementado, más aún ahora que la empresa ha comenzado a desplegar su información en la web para conocimiento de otras instituciones públicas.

Los cambios, modificaciones y nuevas implementaciones al Sistema Comercial son reportados en un su gran mayoría por las áreas usuarias, quienes brindan una primera retro alimentación de la efectividad que muestran las distintas funcionalidades de los módulos. El resto de los casos son estudiados y analizados por los propios desarrolladores quienes ven oportunidades de mejora a los desarrollos de producción.

El Sistema de la Abastecedora es el sistema que administra la lógica de la compañía para la adquisición y contratación de los productos terminados antes de la comercialización. Aquí se administran guías de remisión,

máximos y mínimos de productos por centro de distribución, índices de producción, entre otros módulos. Actualmente, el grupo de desarrolladores de JAVA se encuentra en el rediseño del sistema de la Abastecedora, el mismo que fue desarrollado hace 30 años aproximadamente con herramientas y lenguajes de programación como COBOL y RPG.

Debido a que el rediseño de este sistema está empezando, entonces las herramientas y lenguajes utilizados se encaminan a la programación WEB directamente. Actualmente se utiliza JAVA como el lenguaje primario, y otras variaciones del mismo para desarrollar los módulos con tecnología WebSphere.

- **ELEMENTOS QUE INTERVIENEN EN EL PROCESO DE DESARROLLO DE SISTEMAS EN JAVA**

  3.1.4.1.1. FASES

  - **ANÁLISIS DE REQUERIMIENTOS**

En esta fase, el personal de desarrollo, líder del proyecto, entre otros, realiza la concepción del proyecto de desarrollo mediante el análisis del requerimiento o acatando las distintas disposiciones de los organismos de control como la Dirección Nacional de Hidrocarburos. Para esto, se utilizan formularios simples en los que se registra la información general del requerimiento: módulo a afectar, información que se quiere obtener y definición de la necesidad del requerimiento. Estos requerimientos sirven para identificar el objeto de la creación, modificación o eliminación.

Las áreas usuarias son las que reportan los requerimientos, quienes en primera instancia comunican la necesidad al Jefe de la Unidad vía telefónica. Éste a su vez, se encarga de indicar al usuario que todo requerimiento debe ser solicitado de Jefatura a Jefatura, ya que en la mayoría de los casos son los usuarios finales los que detectan una oportunidad de mejora, conversan con sus superiores pero son ellos mismos quienes reportan los requerimientos; entonces, el Jefe de la Unidad usuaria realiza una solicitud vía memo al Jefe de Sistemas, en donde se resume la necesidad reportada. Allí se indica de la necesidad de llenar el formato de notificación de cambios. Debido a que en muchas ocasiones el área usuaria no tiene tiempo o el formato a la mano, el Jefe de Desarrollo o Líder del Proyecto llena este formato a nombre de él.

Luego, el requerimiento es analizado por el Jefe de Sistemas, Jimmy Murillo (quien cumple la función de Jefe de Proyecto (para todos los casos), la Jefe de Desarrollo, Blanca Rivera (quien cumple la función de Líder de Proyecto), el Analista de Sistemas IIIA, Fernando Tapia (quien se encarga de administrar al equipo de desarrollo y cumple con varios roles, compartiendo inclusive el rol de Líder de Proyecto durante algunas fases del desarrollo), y la Administradora de la Base de Datos, Rosa Romero. Este grupo es conocido informalmente como el comité de cambios.

Luego del análisis acerca de si es una funcionalidad existente o nueva, o de una corrección, se decide la aceptación del requerimiento y se establece el

tiempo de entrega de la solución, si es que es aceptada. Todo esto se registra en la misma acta donde se solicitó el requerimiento para luego archivar el acta. Seguidamente se designa al personal del área que conformará el equipo de desarrollo y, finalmente, se establece la documentación, así como la constitución de los entregables parciales y totales de la solución que deben estar disponible para las fechas acordadas durante el período. Se lleva una bitácora manual de los requerimientos suscitados.

El formato en el que se acepta un requerimiento puede ser de dos formas:

- Fotocopia del requerimiento, sumillada, con la información original (se utiliza memorándums debido a la necesidad de afectar directamente a la aplicación de producción y al tiempo que se necesita para desarrollar esto)

- Correo electrónico, en caso de cambios menores, o actas de reuniones con una sumilla de “Sistemas aplicar”, en el cual se decide que firmen los responsables.

- **DISEÑO**

Actualmente, en esta etapa se establece la forma en la que el sistema cumplirá con los requerimientos identificados durante la fase de análisis, y se identificarán las salidas que debe producir el sistema.

En el diseño, también se especifican los datos de entrada, los datos que serán base para los cálculos y los datos que deben ser almacenados; esto

colabora para que la selección de las estructuras de archivo y los dispositivos de almacenamiento sea más efectiva.

El Líder del Proyecto exige que, como parte del trabajo inicial, los programadores realicen un diseño de la interfaz para evaluar cómo se ésta se va a presentar al usuario final. Sin embargo, esta interfaz se dibuja en papeles sueltos que dispone el desarrollador pero que no forman parte de la documentación formal de todo el proceso de desarrollo. Al final, todos los diseños, o bosquejos anteriores, se quedan en las hojas personales de los desarrolladores, y en general se maneja un conocimiento empírico o basado en la memoria personal, de cómo debe estar diseñada la interfaz y de cómo se debe estructurar la transmisión de la información. Las dimensiones son distintas siempre, aunque las interfaces guarden cierta relación gráfica unas con otras en cuanto a la posición de los objetos.

- **DESARROLLO**

El equipo de desarrollo se encarga de realizar la programación de la solución en ambiente de pruebas. Inicialmente se asigna la responsabilidad de un módulo o de parte de la programación total a cada desarrollador. El siguiente paso es definir la fecha de entrega de cada parte del desarrollo para que el proyecto no sufra demoras en la fecha final de entrega; sin embargo, los retrasos es considerado como parte del desarrollo, por lo que los proyectos tardan mucho hasta obtener la versión piloto final.

Ya en la etapa de desarrollo propiamente, la programación de la solución se realiza progresivamente. Cada desarrollador tiene acceso al ambiente a pruebas tanto para la aplicación, como para la base de datos, y se encarga de ir adicionando código para la estructura y lógica a sus módulos, así como de probar la funcionalidad de dichas adiciones.

Luego de realizar pruebas reiterativas de cada parte del desarrollo, se procede a definir una versión inicial total para hacer una prueba más completa que comprenderá análisis del comportamiento en escenarios críticos que verifiquen su funcionalidad.

La metodología que se sigue para desarrollar es secuencial; es decir, se realizan y prueban rutinas o procedimientos cortos, para luego pasar a la fase de pruebas globales. Cabe destacar que no se genera documentación alguna de las pruebas de desarrollo y que el control de los tiempos de entrega de los productos parciales, se lo hace verbalmente al inicio de semana donde se averigua el estado actual de desarrollo del módulo encargado a cada desarrollador.

- **PRUEBAS**

Antes de pasar a la etapa de pruebas, se tiene que ordenar todas las partes de la solución que han estado a cargo del equipo de desarrollo, para consolidarlas en una versión final. Este proceso de integración es ejecutado por el Analista de Tecnología IIIA, Fernando Tapia, quien es el encargado de producir una versión del sistema para registrarlo en el controlador de

versiones y luego publicarlo en el servidor de pruebas con la finalidad de que el equipo de desarrollo y, eventualmente, el personal del área usuaria realicen las pruebas con información real.

Dependiendo de la complejidad de la solución, se realizan casos de uso en los cuales se especifican los requerimientos de prueba que deben ser satisfechos antes de dar por exitoso el desarrollo de una solución de software. Sin embargo esto no es mandatorio.

Todo esto se documenta en un formato denominado “DESARROLLLO DE SW-PCO-09”, el mismo que es almacenado en una locación específica dentro del servidor, y al cual tiene acceso todo el personal de la Unidad de Sistemas y Telecomunicaciones.

La aprobación de la etapa de pruebas puede ser registrada de tres formas: con una sumilla en el documento físico del requerimiento inicial, por una notificación vía correo electrónico al área usuaria, o de forma verbal por parte del equipo de desarrollo a los usuarios que reportaron el requerimiento. No se tiene una vía única de comunicación.

En el ambiente de pruebas se tiene almacenada información del ambiente de producción de dos meses anteriores como máximo, lo que garantiza que las pruebas utilicen información actualizada para verificar la funcionalidad de la solución.

- **IMPLEMENTACIÓN, EVALAUACIÓN Y MANTENIMIENTO**

Luego de realizar las pruebas sobre los procedimientos del sistema desarrollado, se procede a referenciar el proyecto al ambiente de producción; para esto, la solución es re-direccionada a las librerías correspondientes al ambiente de producción, para luego ser compilada y, finalmente, evaluada en su efectividad por el usuario. Esta función es realizada por el Analista de Tecnología IIIA, Fernando Tapia, en la mayoría de los casos, o por la Jefe de Desarrollo, Blanca Rivera, en ausencia de Fernando, quienes cumplen con la función de compiladores, y Líderes del Proyecto.

Luego se hace un seguimiento durante un par de días en el que se evalúa el comportamiento de la solución. En caso que sea necesario realizar un cambio o que parte de la implementación no sea aceptada, el área usuaria reporta al administrador de la aplicación este hecho vía telefónica o verbalmente para que éste proceda con la realización de las correcciones pertinentes. Este reporte es analizado breve e informalmente por el Analista de Tecnología IIIA, Fernando Tapia. La mayoría de las veces, las correcciones ya no son tratadas como requerimientos nuevos, sino que el ciclo de vida de cada corrección dependerá de lo encontrado por el área usuaria y de qué tan grave es el incumplimiento al requerimiento inicial. Para el caso donde los cambios son menores, se comunica al desarrollador la falla, se corrige a la brevedad la disfuncionalidad (24 a 48 horas máximo), y se comunica al Analista de Tecnología IIIA, para que referencie la solución en el servidor de producción y comunique al usuario la disponibilidad del mismo.

El mantenimiento de la aplicación, sistema o producto de software, se realiza mediante la comprensión de las facetas y procesos inmersos en el desarrollo para definir qué es lo que se hace, cómo se lo hace, y con qué frecuencia se presentan dichas excepciones. No se realiza mantenimiento a las aplicaciones a no ser que colapse un módulo o funcionalidad importante.

Debido a que el registro del requerimiento inicial es muy general, y a que los desarrolladores realizan cambios múltiples y continuos a todos varios sistemas, es común encontrarse con módulos pequeños desarrollados reiterativamente, así como requerimientos que han sido aprobados para su implementación pero que a mitad del proceso de desarrollo son detenidos porque se identifica la solución previamente implementada en el sistema.

3.1.4.1.2. ROLES, CARGOS Y PERSONAS

Del relevamiento realizado se definieron los siguientes roles:

PERSONAS	CARGO	ROLES	ETAPAS
Fernando Tapia	Especialista IIIA	Líder del Proyecto	Todas
		Analista de Sistemas	Análisis, Diseño
		Arquitecto de Software	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Líder de Pruebas	Desarrollo, Pruebas
		Verificador	Pruebas
		Integrador	Desarrollo, Pruebas
		Implementador	Implementación
Roberth Gallo	Especialista IIA	Implementador	Implementación *
		Analista de Sistemas	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador Senior	Desarrollo
Gina Pérez	Especialista IIA	Analista de Sistemas	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador semi Senior	Desarrollo
Blanca Rivera	Especialista IVA	Líder del Proyecto	Todas **
		Implementador	Implementación
Leonor Mera	Especialista IA	Analista de Sistemas	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador	Desarrollo
Sandra Salazar	Especialista IA	Analista de Sistemas	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador	Desarrollo
Rafael Armendariz	Especialista IA	Arquitecto de Software	Análisis, Diseño
		Analista de Sistemas	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador	Desarrollo
Patricio León	Especialista IA	Analista de Sistemas	Análisis, Diseño
		Diseñador de Base de Datos	Diseño, Desarrollo
		Desarrollador	Desarrollo
Área Usuaria	Usuario Final	Verificador	Pruebas

Tabla 3.22. Descripción de los roles, cargos y personas que intervienen en el proceso de desarrollo con JAVA
Realizado por: Xavier Salazar / Eduardo Velalcázar

---
* en reemplazo de Fernando Tapia cuando éste no está
** revisa y coordina los recursos tecnológicos y humanos. Aprueba la planificación de cada proyecto y lleva el control de avance de todos los proyectos de desarrollo

Además de las personas que constan en la tabla, intervienen los Jefes de Área que vienen a ser supervisores del proceso de aprobación. Por ejemplo, si un usuario de la Jefatura Financiera detecta fallas en la solución, el Jede de Área redacta un memorándum en referencia a dicho requerimiento para su análisis.

3.1.4.2. DIAGRAMAS DE FLUJO DEL PROCESO DE DESARROLLO

a) DIAGRAMA GENERAL

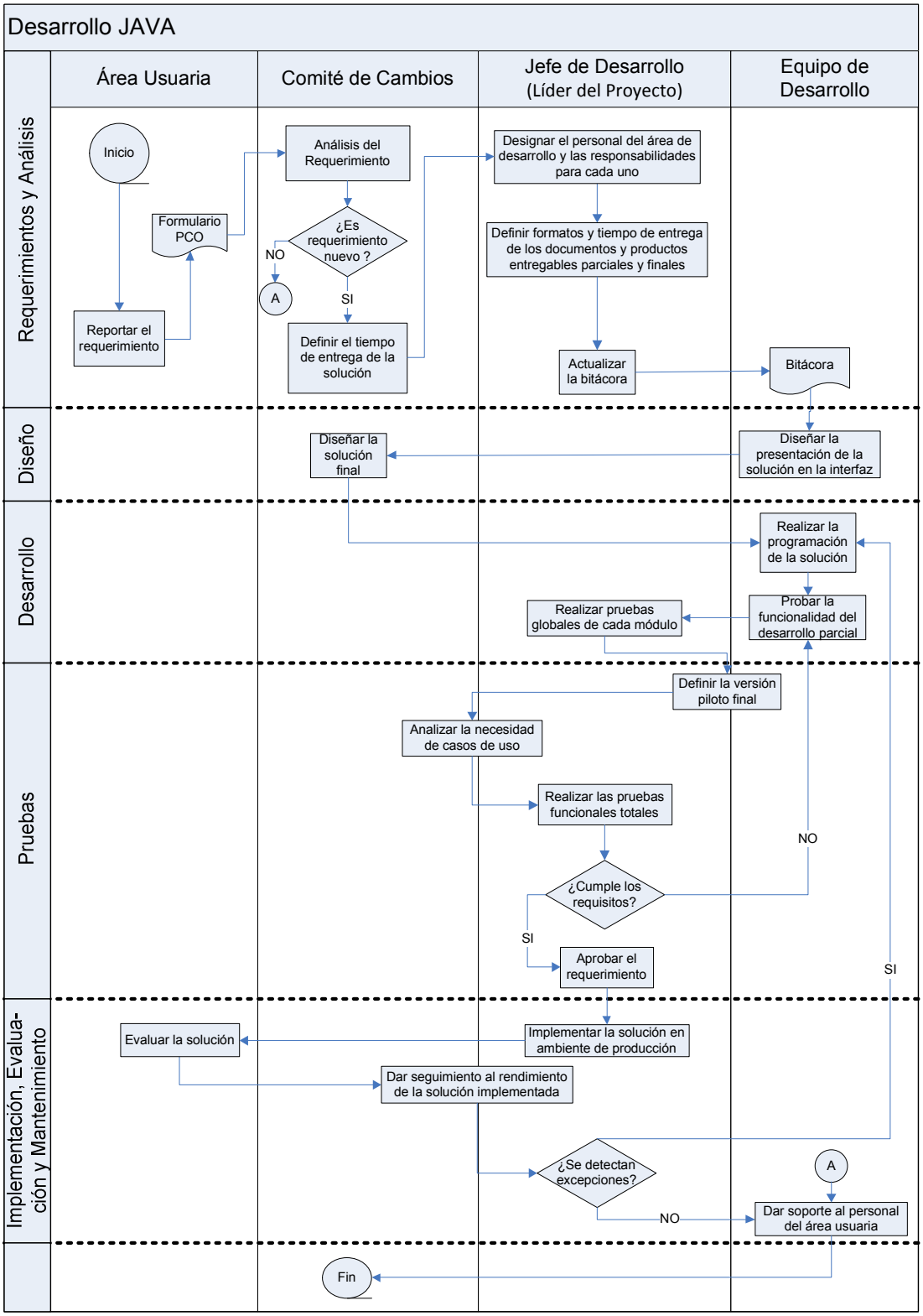
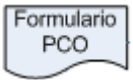
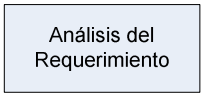
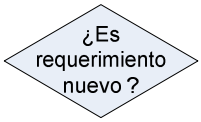
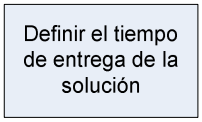
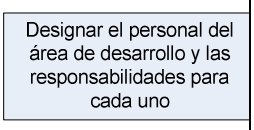
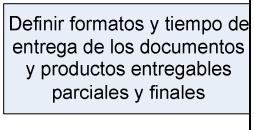


Gráfico 3.18.Flujo General de Procesos para el proceso de desarrollo - JAVA
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS – Diagrama General**

Fase	Elemento	Descripción	Responsable
Requerimientos y Análisis		Documento para registrar información general del requerimiento que usualmente contiene los detalles de la conversación telefónica o verbal con el Jefe de la Unidad o Jefe de Desarrollo.	Área Usuaria / Jefe de Desarrollo
Requerimientos y Análisis		El área usuaria identifica el requerimiento, conversa vía telefónica con el Jefe de Desarrollo o Unidad, acerca de una necesidad específica.	Área Usuaria / Jefe de Unidad (parte del Comité de Cambios)
Requerimientos y Análisis		El Comité de Cambios realiza un análisis de la petición de cambios para luego definir si es factible, si ya se ha implementado o si hace falta dar soporte al usuario (ver el diagrama detallado).	Comité de Cambios
Requerimientos y Análisis		Pregunta que debe ser definida luego del Análisis del Requerimiento. En caso positivo se procede a definir el tiempo de entrega de la solución, de lo contrario se da soporte al usuario.	Comité de Cambios
Requerimientos y Análisis		Se define el tiempo de entrega de la solución de acuerdo a la complejidad del mismo, prioridad, y evaluación de las actividades de los desarrolladores; siempre se estima un tiempo de retraso (ver el diagrama detallado).	Comité de cambios
Requerimientos y Análisis		El Analista de Tecnología IIIA designa al (a los) responsable (s) de desarrollar la solución del requerimiento.	Líder del Proyecto / Jefe de Desarrollo
Requerimientos y Análisis		Luego de designar el personal, el Analista de Tecnología IIIA establece los avances parciales y finales. Incluye una estimación de	Jefe de Desarrollo / Líder del Proyecto

		tiempos de entrega y formatos. Esto se hace informalmente en hojas de cuaderno.	
Requerimientos y Análisis	Actualizar la bitácora	Existe una bitácora para registrar todos los requerimientos suscitados. Esta acción la realiza el Analista de Tecnología IIIA, Fernando Tapia, o la Analista de Tecnología IIA, Gina Pérez.	Jefe de Desarrollo / Líder del Proyecto, Desarrollador senior
Requerimientos y Análisis	Bitácora	Documento o registro físico que se lleva en un cuaderno propiedad del Analista de Sistemas IIIA, Fernando Tapia.	Líder del Proyecto, Desarrollador senior
Diseño	Diseñar la presentación de la solución en la interfaz	El desarrollador designado debe realizar un esquema gráfico y un flujo de datos, utilizando la solución discutida para que sirva de guía en el proceso de diseño y desarrollo. El flujo muestra como viaja la información desde que se ingresa en pantalla hasta que se registra en la base de datos.	Equipo de desarrollo / Desarrollador senior
Diseño	Diseñar la solución final	Se analiza la propuesta del diseño realizado por el Equipo de desarrollo y se ajusta algún detalle que pueda surgir. El criterio de diseño entre el equipo de desarrollo y el comité de cambios es bastante similar, por lo que casi nunca se tienen cambios grandes.	Comité de cambios
Desarrollo	Realizar la programación de la solución	Se da inicio el proceso de desarrollo de la aplicación. Ver el diagrama a detalle más adelante.	Equipo de desarrollo
Desarrollo	Probar la funcionalidad del desarrollo parcial	Se compila el código desarrollado. No se tienen casos de prueba formalmente definidos.	Equipo de desarrollo
Desarrollo	Realizar pruebas globales de cada módulo	Se realizan pruebas del comportamiento de la solución en cada uno de los módulos que son afectados.	Jefe de Desarrollo / Líder del Proyecto

		La solución es desplegada en ambiente de desarrollo.	
Pruebas	Definir la versión piloto final	Es posible que se realicen cambios al código, a discreción del Jefe de Desarrollo, para mejorar el desempeño de la solución. Se supervisa que la versión sea completa y no genere errores.	Jefe de Desarrollo / Líder del Proyecto
Pruebas	Analizar la necesidad de casos de uso	El Jefe de Desarrollo, o el Comité de Cambios (si la solución es muy importante y afecta a muchos módulos), define la necesidad de desarrollar casos de uso: los documenta y procede.	Jefe de Desarrollo / Comité de Cambios
Pruebas	Realizar las pruebas funcionales totales	Esta prueba consiste en ingresar información real, que afecte a más de un módulo, y verificar la consistencia de la solución.	Jefe de Desarrollo / Líder del Proyecto
Pruebas	¿Cumple los requisitos?	Si es afirmativo se aprueba el requerimiento, caso contrario, se prueba otra vez la funcionalidad parcial (realizando cambios menores al código y compilando).	Jefe de Desarrollo / Desarrollador Senior (en algunas ocasiones)
Pruebas	Aprobar el requerimiento	Se da por terminado el requerimiento	Jefe de Desarrollo / Líder del Proyecto
Implementación, Evaluación y Mantenimiento	Implementar la solución en ambiente de producción	El acceso a la biblioteca de producción la tiene el Jefe de Desarrollo desde su máquina. Esta acción es realizada luego de que se ha aprobado el requerimiento y debido a que el área usuaria necesita evaluar que las consideraciones del requerimiento han sido implementadas correctamente	Jefe de Desarrollo / Líder del Proyecto (si no se encuentra el desarrollador semi senior es quien despliega en producción)
Implementación, Evaluación y Mantenimiento	Evaluar la solución	El área usuaria recibe un correo electrónico o una llamada de notificación sobre el cumplimiento del desarrollo del requerimiento para que pruebe las	Área usuaria

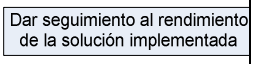
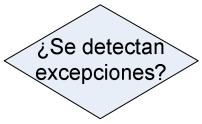
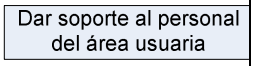
		funcionalidades solicitadas.	
Implementación, Evaluación y Mantenimiento		La evaluación que realiza el Área Usuaria permite que el Jefe de Desarrollo identifique posibles problemas o mejoras a la solución. Además, al dar seguimiento a la implementación, se da soporte al usuario vía telefónica para que identifique las funcionalidades de forma correcta y no reporte excepciones inexistentes	Jefe de Desarrollo / Líder del Proyecto
Implementación, Evaluación y Mantenimiento		Pregunta que debe ser respondida luego de culminada la evaluación del Área Usuaria. En caso positivo se comunica al Equipo de Desarrollo que debe corregir la(s) excepción(es) encontrada(s), caso contrario se da por exitosa la solución	Jefe de Desarrollo / Líder del Proyecto
Implementación, Evaluación y Mantenimiento		El Equipo de Desarrollo se encarga de capacitar al personal del Área Usuaria en la funcionalidad adherida.	Equipo de Desarrollo

Tabla 3.23. Descripción de fases, elementos y responsables del proceso de desarrollo de aplicaciones con JAVA – Diagrama General
Realizado por: Xavier Salazar / Eduardo Velalcázar

b) DIAGRAMAS POR FASE

ANÁLISIS DE REQUERIMIENTOS

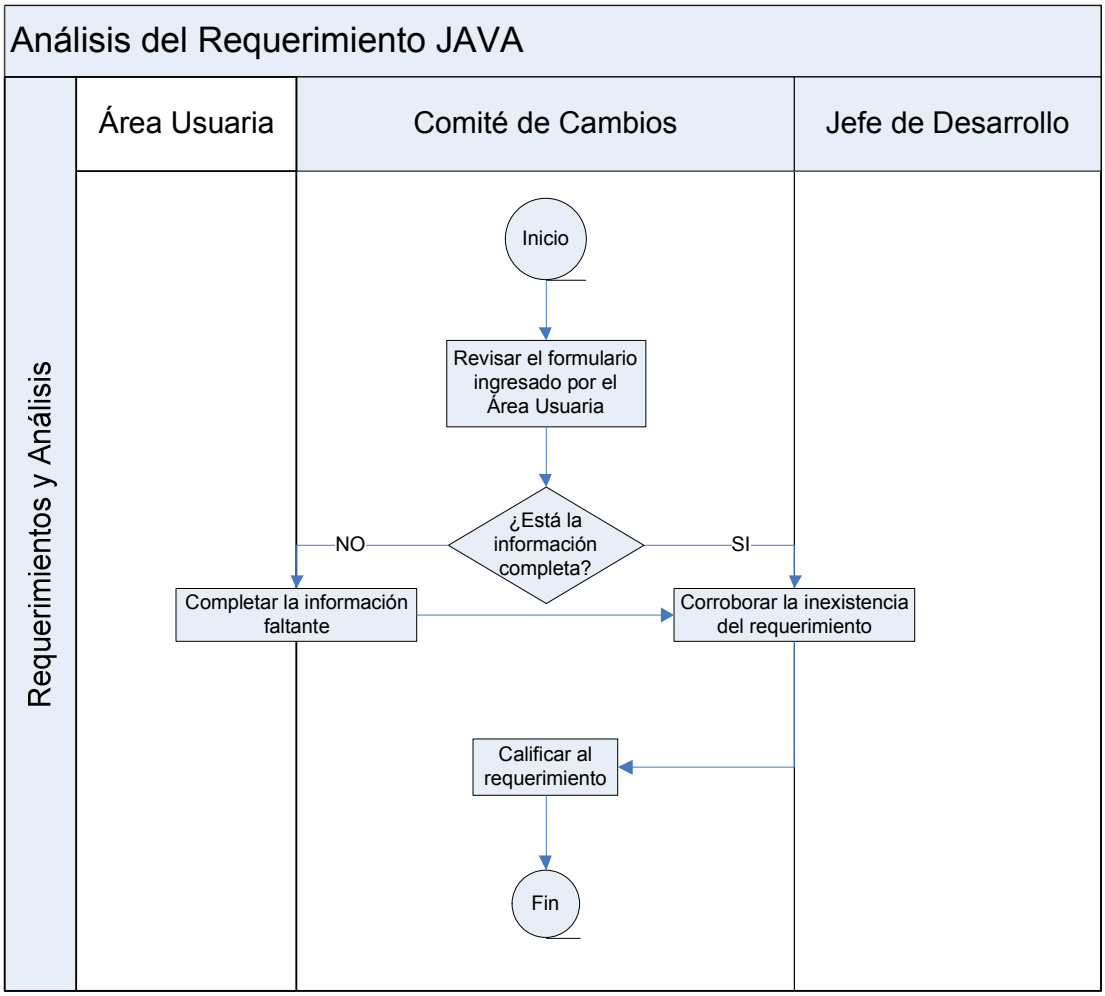


Gráfico 3.19.Flujo de Proceso Análisis de Requerimientos para la fase de Requerimientos y Análisis - JAVA
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS – Análisis de Requerimientos**

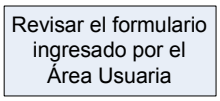
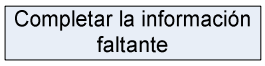
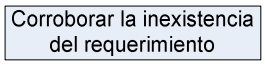
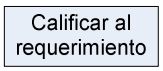
Elemento	Descripción	Responsable
	Los requerimientos son elaborados por las áreas usuarias que trabajan con el Sistema de la Comercializadora. La revisión consiste en que se especifique el módulo del sistema, la acción a realizar y el beneficio.	Comité de Cambios
	Se revisa que la información detalla anteriormente esté completa y que el formulario esté firmado por el Jefe del Área Usuaria.	Comité de Cambios / Jefe de Desarrollo
	En caso de que la información registrada en el formulario esté incompleta, se solicita al Área Usuaria que registre completamente la información antes de procesar dicho formulario.	Área Usuaria / Comité de Cambios
	En caso de que el formulario esté completo, se revisa que la funcionalidad no esté ya implementada en el sistema, con la finalidad de no duplicar esfuerzos. Para esto, el Jefe de Desarrollo asesora al Comité de Cambios.	Comité de Cambios / Jefe de Desarrollo
	La calificación consiste en definir: existencia previa, necesidad, prioridad, entre otros factores relevantes. Este análisis y definición es informal. Al final se registra la palabra aceptado / no aceptado en el formulario. Sin embargo, no se detalla las especificaciones del análisis.	Comité de Cambios

Tabla 3.24. Descripción de elementos y responsables en el proceso de Análisis de Requerimientos con JAVA – Fase Requerimientos y Análisis
Realizado por: Xavier Salazar / Eduardo Velalcázar

DEFINIR EL TIEMPO DE ENTREGA DE LA SOLUCIÓN

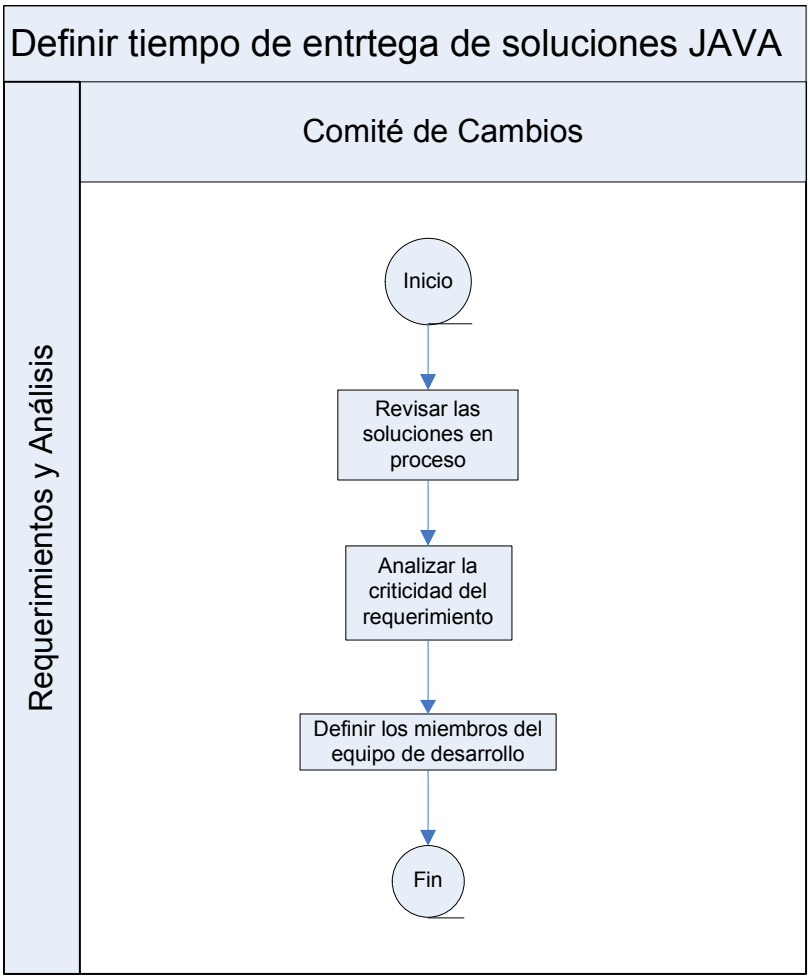


Gráfico 3.20.Flujo de Proceso Definir Tiempo de Entrega de Soluciones para la fase de Requerimientos y Análisis - JAVA
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS – Análisis de Requerimientos**

Elemento	Descripción	Responsable
Revisar las soluciones en proceso	Se mantiene un cuadro de control de actividades donde se verifica el estado de cada solución en proceso. Este cuadro es responsabilidad de la Jefe del Área de Desarrollo, quien es parte del Comité de Cambios.	Comité de Cambios / Jefa del Área de Desarrollo
Analizar la criticidad del requerimiento	Se incluye el requerimiento en la lista de soluciones en proceso y se reorganiza las prioridades.	Comité de Cambios
Definir los miembros del equipo de desarrollo	Se tiene el control de las actividades que realiza cada programador. En base a la experiencia y disponibilidad de cada uno, se define a los miembros del equipo de desarrollo para la solución.	Comité de Cambios / Jefe de Desarrollo

Tabla 3.25. Descripción de elementos y responsables en el proceso de Definir Tiempo de Entrega de Soluciones con JAVA – Fase Requerimientos y Análisis
Realizado por: Xavier Salazar / Eduardo Velalcázar

REALIZAR LA PROGRAMACIÓN DE LA SOLUCIÓN

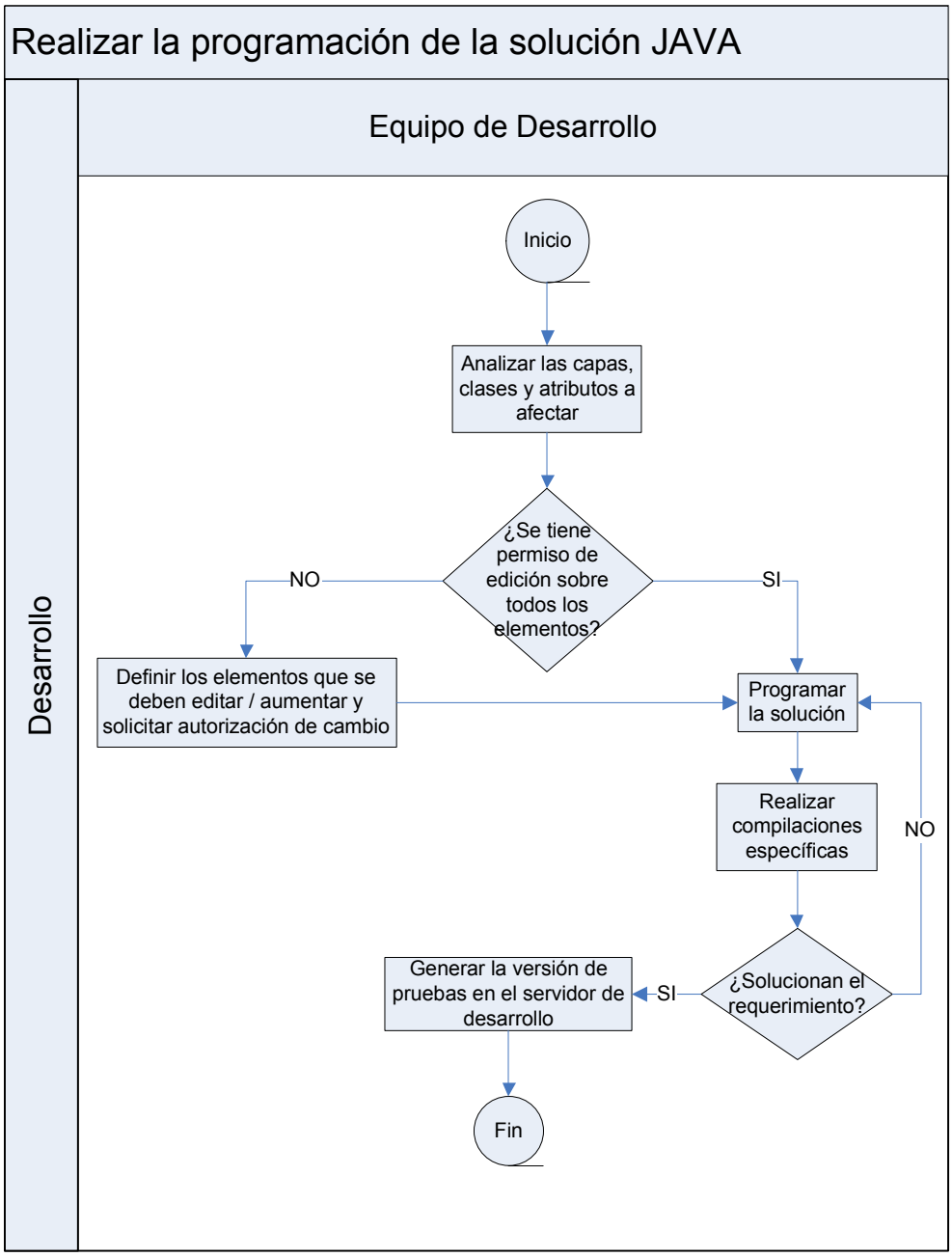
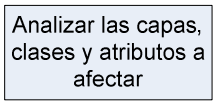
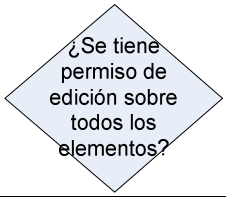
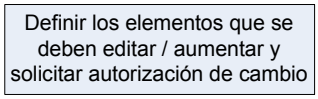
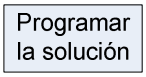
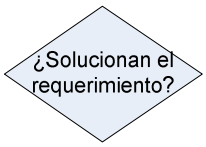


Gráfico 3.21.Flujo de Procesos para la fase de Desarrollo - JAVA
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS - Desarrollo**

Elemento	Descripción	Responsables
	Se define claramente todas las capas, clases y elementos que se verán afectados con la implementación de la solución. Este análisis se lo realiza informalmente en los cuadernos físicos pertenecientes a los miembros del equipo de desarrollo y no cumplen con ningún formato.	Equipo de Desarrollo / Jefe de Desarrollo
	Existen elementos (usualmente son clases) que por su criticidad son inaccesibles por cualquier programador; inclusive en ambiente de desarrollo.	Equipo de Desarrollo / Jefe de Desarrollo
	Se realiza una petición verbal al Jefe de Desarrollo para que añada el usuario que pertenece al desarrollador, al grupo de administradores de la(s) clase(s) y proceder con la programación. El permiso es temporal y al culminar la implementación de la solución, el Jefe de Desarrollo depura la lista nuevamente.	Equipo de Desarrollo / Jefe de Desarrollo
	La programación se realiza en base a la experiencia de cada programador. Existe código que se recicla de otras clases; sin embargo no existe una guía formal de las funcionalidades similares, lo que potencia la heterogeneidad de códigos para un mismo fin.	Equipo de Desarrollo
	Como en todo proceso de desarrollo, se va modificando el código fuente y se compilan los cambios.	Equipo de Desarrollo
	Si dichos cambios consiguen cumplir con las especificaciones, se genera la versión de prueba final; caso contrario, se regresa al proceso de programación de la solución nuevamente.	Equipo de Desarrollo

Generar la versión de pruebas en el servidor de desarrollo	Una vez que culmina la programación (a opinión del programador) se genera una versión para poder referenciarla en el servidor de desarrollo, previamente a la realización de pruebas sobre el sistema	Equipo de Desarrollo

Tabla 3.26. Descripción de fases, elementos y responsables en el proceso de Realizar la programación de la solución con JAVA – Fase Desarrollo
Realizado por: Xavier Salazar / Eduardo Velalcázar

CAPITULO IV

DESARROLLO DE GUÍA DE PROCESOS DE DESARROLLO

4.1. PROPUESTA PARA LOS SISTEMAS SECUENCIALES: COBOL, RPG Y LOTUS NOTES

4.1.1. ESTRUCTURA Y CONCEPTUALIZACIÓN DE LAS FASES RUP EN EL PROCESO DE DESARROLLO DE APLICACIONES EN RPG, COBOL y LOTUS NOTES

Incepción

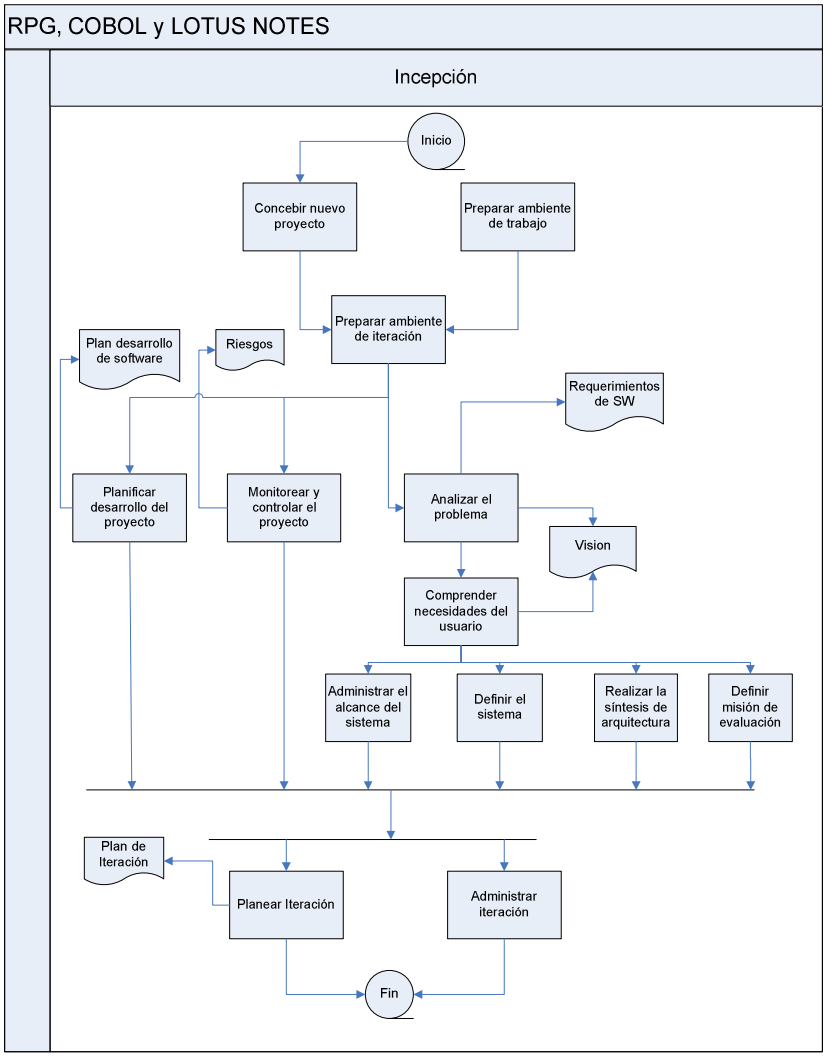


Gráfico 4.1 Flujo de Procesos para la fase de Incepción- RPG, COBOL y LOTUS NOTES
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS RPG, COBOL y LOTUS NOTES**

Incepción

Elemento	Descripción	Responsable	Roles
Concebir nuevo proyecto	Desarrollar un plan económico para realizar la visión del proyecto, decidir si se continua o abandona el desarrollo de un nuevo proyecto	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Preparar ambiente de trabajo	Prescribir y describir la dirección del proceso de desarrollo	Administrador RPG, COBOL y LOTUS	Ingeniero Proceso/Especialista Herramienta
Preparar ambiente de iteración	- Asegurar que el administrador de la aplicación este apropiadamente enterado en el desarrollo del proceso - LLevar una dirección de como modificar los procesos de desarrollo, para ajustarlo a las necesidades del proyecto	Administrador RPG, COBOL y LOTUS	Ingeniero Proceso/Especialista Herramienta
Planificar desarrollo del proyecto	Desarrollar los componentes y los anexos del Plan de Desarrollo de Software	Revisor Administración del Proyecto/ Administrador	Administrador Proyecto/Revisor administración
Plan desarrollo de software	Documento que obtiene información requerida para administrar y llevar un control del proyecto	Administrador del Proyecto	Administrador Proyecto
Monitorear y controlar el proyecto	Monitorear el trabajo y el estado del proyecto, tratar dificultades	Administrador RPG, COBOL y LOTUS / Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Riesgos	Lista de riesgos identificados en el desarrollo del proyecto, en la lista se ordenan por criticidad para poder mitigarlos	Administrador RPG, COBOL y LOTUS	Administrador Proyecto
Analizar el problema	Analizar las necesidades del usuario de PCE, identificar limitaciones impuestas en el sistema	Administrador RPG, COBOL y LOTUS	Analista del Sistema

Vision	Explica la visión del desarrollo y características de las necesidades del usuario de PCE Contiene las bases para identificar requerimientos técnicos	Administrador RPG, COBOL y LOTUS	Administrador Proyecto
Comprender necesidades del usuario	Entender lo requerido por el usuario PCO, obteniendo información del producto deseado	Administrador RPG, COBOL y LOTUS	Analista del Sistema
Administrar el alcance del sistema	Elaborar el alcance del desarrollo del proyecto lo más explícito posible	Administrador RPG, COBOL y LOTUS	Arquitecto Software/Analista de Sistema
Definir el sistema	Converger el alcance del desarrollo del proyecto a un nivel alto de requerimientos	Administrador RPG, COBOL y LOTUS	Analista del Sistema
Realizar la síntesis de arquitectura	Crear diseño de la arquitectura del software(diseño del proyecto)	Administrador RPG, COBOL y LOTUS	Arquitecto Software/Diseñador
Definir misión de evaluación	Identificar el enfoque del resultado de las pruebas a efectuarse en las iteraciones	Administrador RPG, COBOL y LOTUS	Administrador de Pruebas/Analista de Pruebas/Diseñador de Pruebas
Planear Iteración	Crear un plan de iteración, para generar ayuda en la siguiente iteración	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Plan de Iteración	Conjunto de actividades con recursos asignados; contenido de tareas en el proceso de iteración	Administrador RPG, COBOL y LOTUS	Administrador Proyecto
Administrar iteración	- Actividades de inicio, revisión y fin de iteraciones - Identificar todos los recursos para realizar las iteraciones en la aplicación de RRHH	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración

Tabla 4.1 Descripción de fases, elementos, responsables y roles en el proceso de desarrollo de aplicaciones con RPG, COBOL y LOTUS NOTES – Fase Incepción
Realizado por: Xavier Salazar / Eduardo Velalcázar

Elaboración

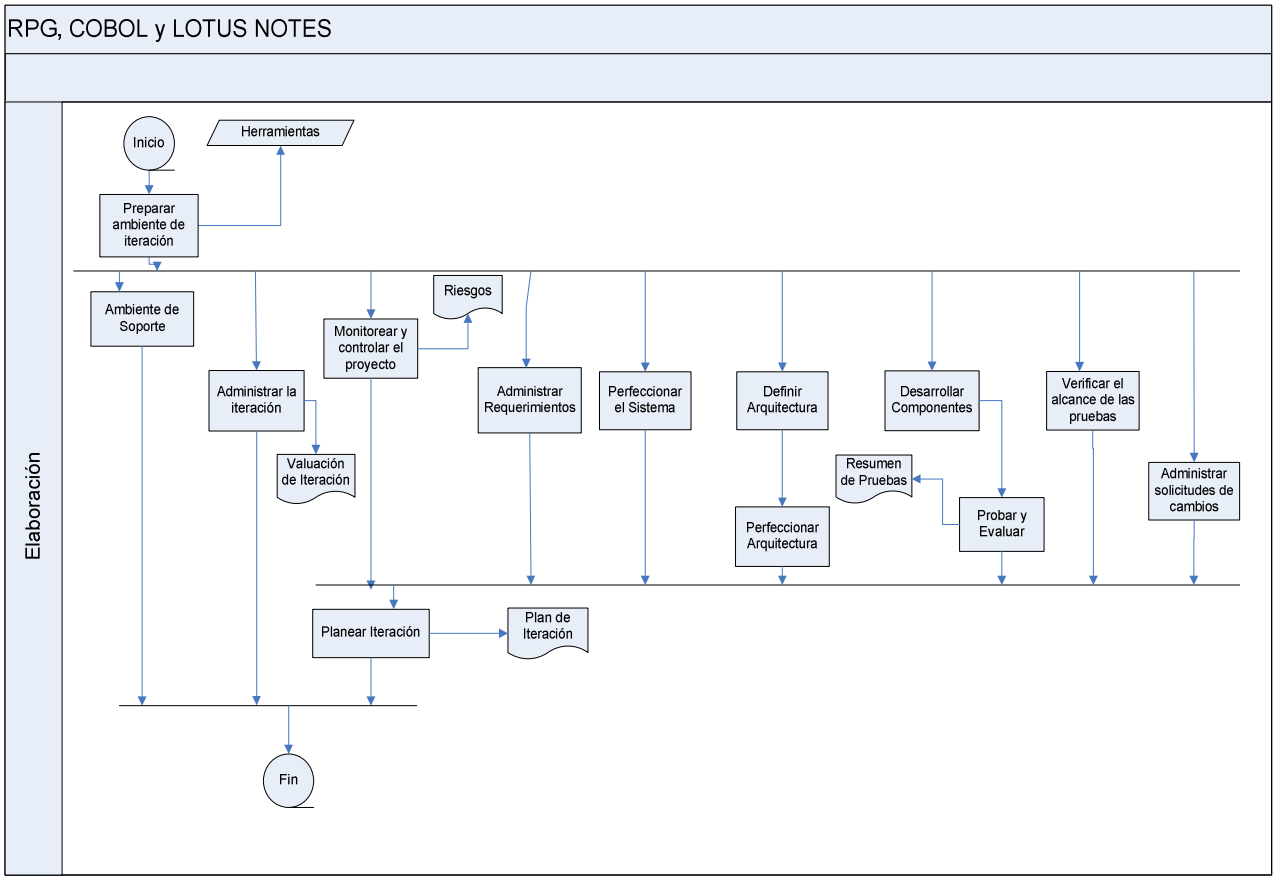


Gráfico 4.2 Flujo de Procesos para la fase de Elaboración- RPG, COBOL y LOTUS NOTES
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS RPG, COBOL y LOTUS NOTES**

Elaboración

Elemento	Descripción	Responsable	Roles
Preparar ambiente de iteración	- Asegurar que el administrador de la aplicación este apropiadamente enterado en el desarrollo del proceso - LLevar una dirección de como modificar los procesos de desarrollo, para ajustarlo a las necesidades del proyecto	Administrador RPG, COBOL y LOTUS	Ingeniero Proceso/Especialista Herramienta
Herramientas	Herramientas utilizadas en el rendimiento del desarrollo de software (Lenguaje RPG)	Administrador RPG, COBOL y LOTUS	Especialista Herramientas
Ambiente de Soporte	Dar soporte al desarrollador de la aplicación RPG, en el uso de herramientas y procesos de una iteración	Administrador RPG, COBOL y LOTUS	Administrador Sistema
Administrar la iteración	Actividades de inicio, revisión y fin de iteraciones - Identificar todos los recursos para realizar las iteraciones en la aplicación de RRHH	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Valuación de Iteración	Captura el resultado de una iteración, lecciones aprendidas y cambios a ser realizados	Administrador RPG, COBOL y LOTUS	Administrador proyecto
Monitorear y controlar el proyecto	Monitorear el trabajo y el estado del proyecto, tratar dificultades	Administrador RPG, COBOL y LOTUS / Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Riesgos	Lista de riesgos identificados en el desarrollo del proyecto, en la lista se ordenan por criticidad para	Administrador RPG, COBOL y LOTUS	Administrador Proyecto

	poder mitigarlos		
Administrar Requerimientos	Evaluar con el usuario PCE el impacto de los cambios requeridos en la aplicación	Administrador RPG, COBOL y LOTUS	Administrador Sistema/Revisor Técnico
Perfeccionar el Sistema	Refinar las necesidades a futuro del usuario de PCE , para entender la definición del proyecto	Administrador RPG, COBOL y LOTUS	Especificador de requerimientos
Definir Arquitectura	Crear diseño de la arquitectura del software(diseño del proyecto)	Administrador RPG, COBOL y LOTUS	Arquitecto de Software/Designer
Perfeccionar Arquitectura	Refinar las necesidades a futuro del usuario de PCE , para entender el diseño del proyecto	Administrador RPG, COBOL y LOTUS	Arquitecto de Software/Revisor Técnico
Desarrollar Componentes	Analizar comportamientos de requerimientos de sw Diseñar el modelo Diseñar componentes de desarrollo Implementar componentes Probar y evaluar componentes Integrar componentes	Administrador RPG, COBOL y LOTUS	Diseñador/ Implementador/ Integrador/ Arquitecto de Software/Revisor Técnico Analista de Pruebas/Probador/ Diseñador de Prueba
Probar y Evaluar	Obtener una apropiada profundidad y anchura del rendimiento de las pruebas efectuadas Medir el trabajo táctico del desarrollo de las pruebas y evaluación	Administrador RPG, COBOL y LOTUS	Analista de Pruebas/Probador/ Diseñador de Prueba
Verificar el alcance de las pruebas	Demostrar que el alcance de pruebas, ayudarán a desarrollar las pruebas planificadas. Entender limitaciones de las técnicas aplicadas en el desarrollo de la aplicación RPG	Administrador RPG, COBOL y LOTUS	Analista de Pruebas/Probador/ Diseñador de Prueba/Administrador de pruebas
Administrar solicitudes de cambios	Analizar el impacto del cambio a realizarse, en caso de ser alto	Administrador RPG, COBOL y LOTUS	Administrador Proyecto/Administrador Control de

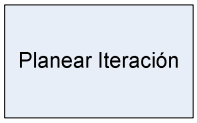
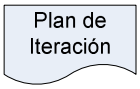
	impacto comunicar al Jefe de desarrollo		Cambios/Analista de pruebas
	Crear un plan de iteración, para generar ayuda en la siguiente iteración	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	
	Conjunto de actividades con recursos asignados; contenido de tareas en el proceso de iteración	Administrador RPG, COBOL y LOTUS	Administrador Proyecto

Tabla 4.2 Descripción de fases, elementos, responsables y roles en el proceso de desarrollo de aplicaciones con RPG, COBOL y LOTUS NOTES – *Fase Elaboración*
Realizado por: Xavier Salazar / Eduardo Velalcázar

Construcción

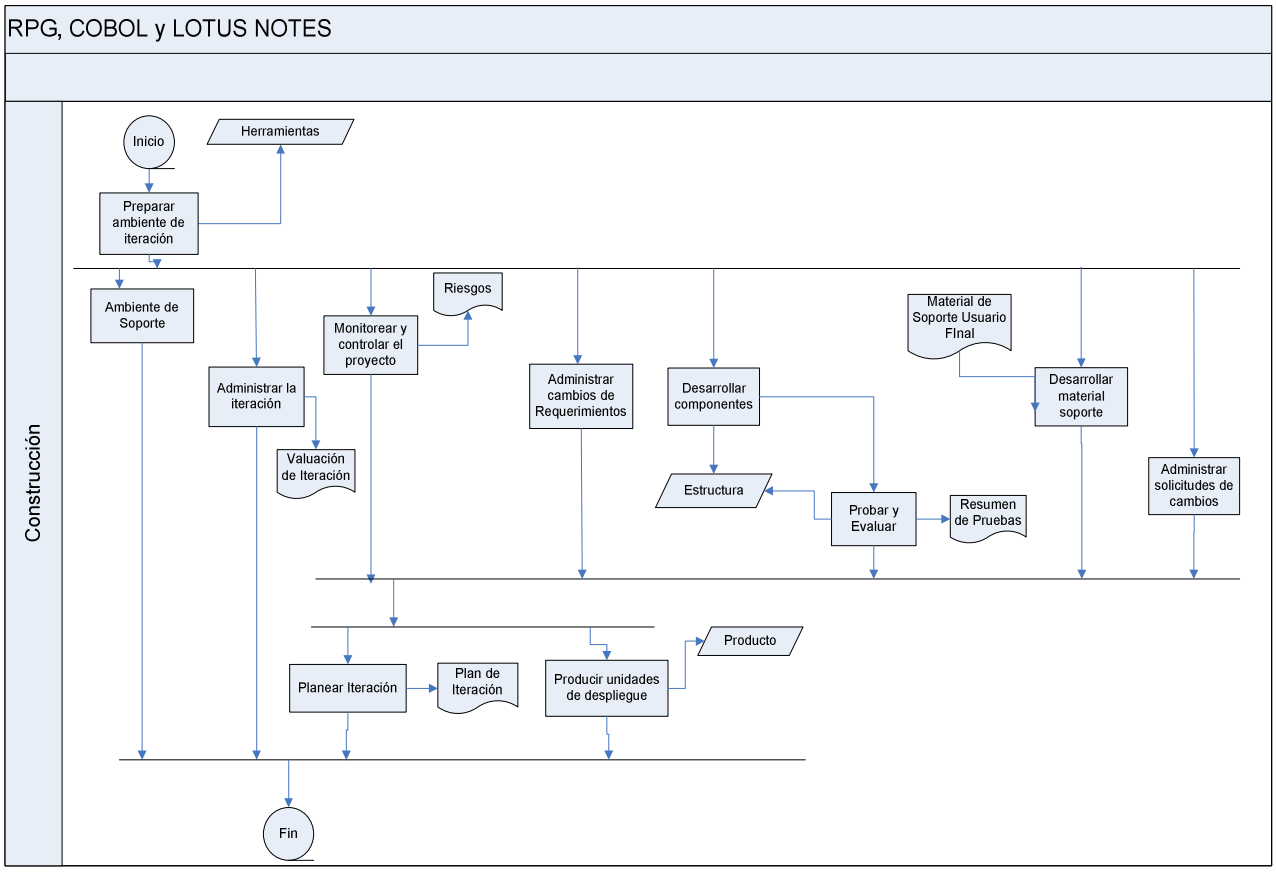


Gráfico 4.3 Flujo de Procesos para la fase de Construcción - RPG, COBOL y LOTUS NOTES
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS RPG Construcción**

Elemento	Descripción	Responsable	Roles
Preparar ambiente de iteración	- Asegurar que el administrador de la aplicación este apropiadamente enterado en el desarrollo del proceso - LLevar una dirección de como modificar los procesos de desarrollo, para ajustarlo a las necesidades del proyecto	Administrador RPG, COBOL y LOTUS	Ingeniero Proceso/ Especialista Herramienta
Herramientas	Herramientas utilizadas en el rendimiento del desarrollo de software (Lenguaje RPG)	Administrador RPG, COBOL y LOTUS	Especialista Herramientas
Soportar el ambiente	Dar soporte al desarrollador de la aplicación RPG, en el uso de	Administrador RPG, COBOL	Administrador Sistema

	herramientas y procesos de una iteración	y LOTUS	
Administrar la iteración	Actividades de inicio, revisión y fin de iteraciones - Identificar todos los recursos para realizar las iteraciones en la aplicación de RRHH	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Valuación de Iteración	Captura el resultado de una iteración, lecciones aprendidas y cambios a ser realizados	Administrador RPG, COBOL y LOTUS	Administrador proyecto
Monitorear y controlar el proyecto	Monitorear el trabajo y el estado del proyecto, tratar dificultades	Administrador RPG, COBOL y LOTUS / Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Riesgos	Lista de riesgos identificados en el desarrollo del proyecto, en la lista se ordenan por criticidad para poder mitigarlos	Administrador RPG, COBOL y LOTUS	Administrador Proyecto
Administrar cambios de Requerimientos	Evaluar con el usuario PCE el impacto de las solicitudes Estructurar el modelo de casos de uso Verificar que los resultados del trabajo de los requerimientos satisfacen necesidades del cliente	Administrador RPG, COBOL y LOTUS	Analista de Sistemas/Revisor Técnico
Desarrollar componentes	Implementar el modelo de desarrollo, siguiendo un proceso iterativo e incremental. Este es un modelo recurrente en las iteraciones de las fases de elaboración y construcción; este modelo debe ser repetido por cada estructura en una iteración, esto se aplica en proyectos que pueden implementarse nuevos módulo dependiendo de los requerimientos de usuarios de RR.HH	Administrador RPG, COBOL y LOTUS	Desarrolladores/Probadores
Probar y Evaluar	Obtener una apropiada profundidad y anchura del rendimiento de las pruebas efectuadas Medir el trabajo táctico del desarrollo de las pruebas y evaluación	Administrador RPG, COBOL y LOTUS	Analista de Pruebas/Probador/ Diseñador de Prueba

Desarrollar material soporte	Elaborar necesidades colaterales en caso de falla al desarrollo del proyecto, así producir un producto efectivo	Administrador RPG, COBOL y LOTUS	Desarrollador/ Escritor técnico
Material de Soporte Usuario FInal	El área usuaria recibe manuales que proporcionan el manejo de las funciones de la nueva aplicación por pedido del área de contraloría	Administrador RPG, COBOL y LOTUS	Escritor técnico
Administrar solicitudes de cambios	Analizar el impacto del cambio a realizarse, en caso de ser alto impacto comunicar al Jefe de desarrollo	Administrador RPG, COBOL y LOTUS	Administrador Proyecto/Administrador Control de Cambios/Analista de pruebas
Planear Iteración	Crear un plan de iteración, para generar ayuda en la siguiente iteración	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	
Plan de Iteración	Conjunto de actividades con recursos asignados; contenido de tareas en el proceso de iteración	Administrador RPG, COBOL y LOTUS	Administrador Proyecto
Producir unidades de despliegue	Crear una unidad de despliegue que habilite el proyecto desarrollado para ser correctamente instalado y operado Formato de pruebas	Administrador RPG, COBOL y LOTUS /Jefe de Desarrollo	Administrador de Instalación/Implementador/ Administrador de Configuración
Producto	Contenedor de varias unidades de despliegue Producto digital que contiene el desarrollo del proyecto en RPG	Administrador RPG, COBOL y LOTUS	Administrador de Instalación

Tabla 4.3 Descripción de fases, elementos, responsables y roles en el proceso de desarrollo de aplicaciones con RPG, COBOL y LOTUS NOTES – *Fase Construcción*
Realizado por: Xavier Salazar / Eduardo Velalcázar

Transición

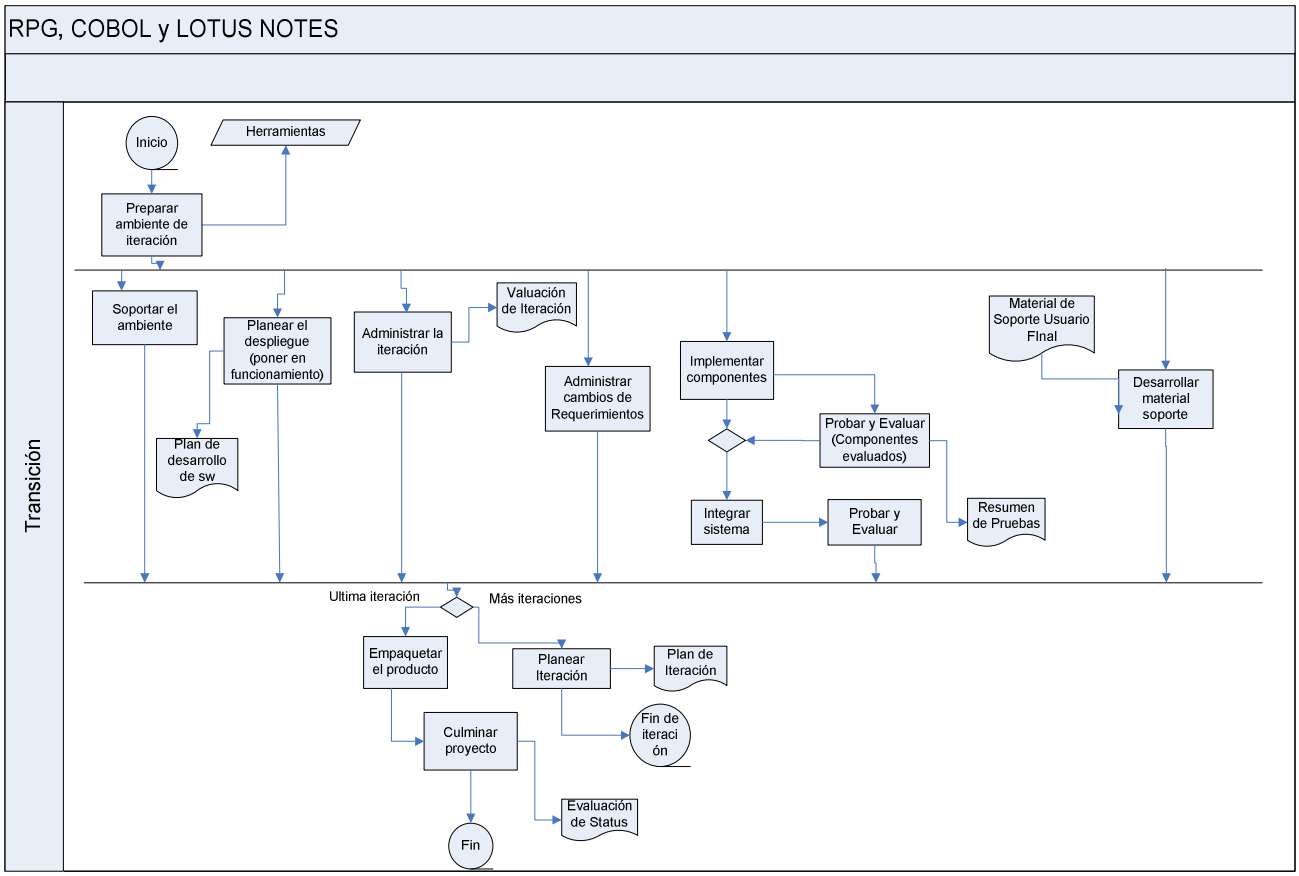


Gráfico 4.4 Flujo de Procesos para la fase de transición - RPG, COBOL y LOTUS NOTES
Realizado por: Xavier Salazar / Eduardo Velalcázar

- **PROCESOS Y DOCUMENTOS RPG, COBOL y LOTUS NOTES**

Transición

Elemento	Descripción	Responsable	Roles
Preparar ambiente de iteración	- Asegurar que el administrador de la aplicación este apropiadamente enterado en el desarrollo del proceso - LLevar una dirección de como modificar los procesos de desarrollo, para ajustarlo a las necesidades del proyecto	Administrador RPG, COBOL y LOTUS	Ingeniero Proceso/ Especialista Herramienta
Herramientas	Herramientas utilizadas en el rendimiento del desarrollo de software (Lenguaje RPG)	Administrador RPG, COBOL y LOTUS	Especialista Herramientas
Soportar el ambiente	Dar soporte al desarrollador de la aplicación RPG, en el uso de herramientas y procesos de una iteración	Administrador RPG, COBOL y LOTUS	Administrador Sistema
Planear la instalación (poner en funcionamiento)	Planear el cómo de poner en funcionamiento (instalar) el producto Planificar cuando y como el usuario final tendrá habilitado el requerimiento solicitado al administrador de RPG	Administrador RPG, COBOL y LOTUS	Administrador instalación
Plan de desarrollo de sw	El Plan de Desarrollo de Software es un resultado tangible, comprensivo y compuesto que junta toda la información requerida para manejar el proyecto. Pla de desarrollo, obedece estándares de PCO, actualizado de acuerdo a las mejores prácticas, desarrollado de acuerdo al diseño y requerimientos	Administrador RPG, COBOL y LOTUS	Administrador del proyecto
Administrar la iteración	Actividades de inicio, revisión y fin de iteraciones - Identificar todos los recursos para realizar las iteraciones en la aplicación de RRHH	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración

Valuación de Iteración	Captura el resultado de una iteración, lecciones aprendidas y cambios a ser realizados	Administrador RPG, COBOL y LOTUS	Administrador proyecto
Administrar cambios de Requerimientos	Evaluar con el usuario PCE el impacto de las solicitudes Estructurar el modelo de casos de uso Verificar que los resultados del trabajo de los requerimientos satisfacen necesidades del cliente	Administrador RPG, COBOL y LOTUS	Analista de Sistemas/Revisor Técnico
Implementar componentes	Completar una parte de la implementación para entregar resultados en la integración del producto, unificar módulos	Administrador RPG, COBOL y LOTUS	Implementador/Integrador/Revisor Técnico
Probar y Evaluar (Componentes evaluados)	Obtener una apropiada profundidad y anchura del rendimiento de las pruebas efectuadas Medir el trabajo táctico del desarrollo de las pruebas y evaluación	Administrador RPG, COBOL y LOTUS	Probador/Analista de Pruebas/Diseñador de Pruebas
Resumen de Pruebas	Resumen del análisis de los resultados de las pruebas Medidas de revisión y evaluación de las pruebas	Administrador RPG, COBOL y LOTUS	Administrador de Pruebas
Integrar sistema	Integrar módulos, para consolidar en un producto final	Administrador RPG, COBOL y LOTUS	Integrador
Probar y Evaluar	Obtener una apropiada profundidad y anchura del rendimiento de las pruebas efectuadas Medir el trabajo táctico del desarrollo de las pruebas y evaluación	Administrador RPG, COBOL y LOTUS	Analista de Pruebas/Probador/ Diseñador de Prueba
Desarrollar material soporte	Elaborar necesidades colaterales en caso de falla al desarrollo del proyecto, así producir un producto efectivo	Administrador RPG, COBOL y LOTUS	Desarrollador/ Escritor técnico
Material de Soporte Usuario FInal	El área usuaria recibe manuales que proporcionan el manejo de las funciones de la nueva aplicación por pedido del área de contraloría	Administrador RPG, COBOL y LOTUS	Escritor técnico

Planear Iteración	Crear un plan de iteración, para generar ayuda en la siguiente iteración	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	
Plan de Iteración	Conjunto de actividades con recursos asignados; contenido de tareas en el proceso de iteración	Administrador RPG, COBOL y LOTUS	Administrador Proyecto
Empaquetar el producto	Tomar la unidad de despliegue, scripts de instalación, manuales de usuario, unificar todo como un producto dirigido al usuario que lo solicitó	Administrador RPG, COBOL y LOTUS	Administrador de Instalación
Culminar proyecto	El administrador del proyecto prepara la revisión que indica la terminación del proyecto	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto/Revisor administración
Evaluación de Status	Documento que provee el mecanismo para asegurar que se hayan cumplido las expectativas de todas las partes durante el ciclo de vida del proyecto	Administrador RPG, COBOL y LOTUS /Revisor Administración del Proyecto	Administrador Proyecto

Tabla 4.4 Descripción de fases, elementos, responsables y roles en el proceso de desarrollo de aplicaciones con RPG, COBOL y LOTUS NOTES – *Fase Transición*
Realizado por: Xavier Salazar / Eduardo Velalcázar

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

La metodología RUP analiza robustamente la situación actual. El análisis está parametrizado por variables locales que reflejan la realidad de la empresa; es decir, la adaptación total de la metodología radica en documentar los actores, roles, actividades y productos de la situación actual para delimitar el alcance de la propuesta en miras hacia el escenario ideal. En este proceso se encuentran, determinan y documentan debilidades o posibles procesos que comprometen la seguridad y calidad de los productos; sin embargo, estas limitantes son realidades de las compañías que no pueden ser modificadas o alteradas con una propuesta, debido a que en ocasiones y, especialmente en el caso analizado (por tratarse de una entidad pública), el personal, las actividades y otros factores, son sujetos a políticas públicas o estatales.

Además, en RUP se desarrollan documentos que definen explícitamente las actividades y roles de un actor, por lo que a la empresa, una vez adaptada la metodología, se le da a conocer las inseguridades potenciales para que sean consideradas en la toma de decisiones al corto, medio y largo plazo. De esta forma el análisis es objetivo, similar al hecho de mirar una fotografía, donde uno puede definir que en “aquel” tiempo, debido a “ciertas” circunstancias, la

estructura de trabajo respondía a “esos” procedimientos. Esto mejora notablemente el ambiente previo a la toma de decisiones o la realización de un nuevo y mejorado plan de desarrollo. Una vez definida ésta preconclusión general, llegamos a las siguientes conclusiones particulares:

- **RPG, COBOL, COBOL Cics y LOTUS:**
  - Se concluye que el número óptimo de iteraciones de desarrollo para todos estos lenguajes es UNO solo por cada fase, debido a que el número de empleados que forman parte del equipo de desarrollo varía entre una a tres personas.

  - El o la encargado(a) de todo el proceso de desarrollo cumple con varios roles y actividades, por lo que la generación de documentación no es precisa ni completa, en muchos casos. Esto no retarda las metas ni dilata los tiempos de entrega necesariamente; sin embargo, la informalidad en todo el proceso no permite visualizar de forma completa ni tampoco administrar correctamente los recursos. La propuesta sugiere la generación de mayor documentación que avale las actividades de cada rol desempeñado.

  - La metodología RUP tiene un enfoque que podría funcionar mejor para lenguajes de programación de tercera y cuarta generación; sin embargo, la generación de formatos y

documentación facilita el desarrollo de nuevos módulos y, por supuesto, la migración del sistema hacia tecnologías web (intenciones actuales de la empresa).

- **JAVA:**
  - Se concluye que el número óptimo de iteraciones de desarrollo para este lenguaje es de DOS, debido a que el número de empleados asignados para este equipo de desarrollo varía entre 5 a 8 personas. Además, se determinó que existe una mayor carga de personal para la fuerza de desarrollo que para las áreas de control.

  - Si bien el o la Jefe de Desarrollo cumple con algunos roles, la responsabilidad de generar la documentación de cada rol recae en otros empleados, por lo que se concluye que no existe una relación clara entre actores, actividades y productos entre los miembros del equipo, lo cual potencializa el cometimiento de errores en los productos a desarrollar y una sobre carga inadecuada de funciones para ciertos roles de control.

  - Se evidenció cierto porcentaje de efectividad en la implementación de metodologías de desarrollo similares a RUP; sin embargo, se determinó que la generación de documentación es un proceso que se lo lleva informalmente (o no se realiza), por lo que el conocimiento de las metodologías no es explícito y

responde, más bien, a destrezas empíricas del personal más experimentado o más antiguo.

5.2. RECOMENDACIONES

- **RPG, COBOL, COBOL Cics y LOTUS:**
  - Considerar la migración de la información de todos los módulos a un nuevo sistema que incluya tecnologías web, con lo que la guía de procesos serviría para desarrollar los módulos de forma más efectiva. Esta necesidad está presente en los objetivos de la empresa debido a la unificación y exposición pública de la información y la política actual de compartir información entre todas las carteras del estado con la finalidad de no replicar esfuerzos, por lo que la recomendación es hacia la generación efectiva la de documentación propuesta por la guía.

  - Se recomienda que los varios roles de control estén bajo la responsabilidad y conocimiento de la Jefatura de Sección e inclusive bajo la Jefatura de Unidad, ya que esto permitiría un mayor conocimiento de las necesidades, debilidades y oportunidades de mejora a la parte administrativa del Departamento, la misma que se encarga de la toma de decisiones.

  - Se sugiere la creación de un proyecto de desarrollo en el software controlador de versiones y la compra o desarrollo de un sistema para control documental, que permita verificar periódicamente los documentos en cola y los proyectos que se van generando.

- **JAVA:**
  - Se recomienda generar una subdivisión en la estructura del equipo de desarrollo, específicamente en los desarrolladores, para que las funciones y roles de supervisión al código programado pueda ser atendido y mejorado a ese mismo nivel, con lo que la inclusión de errores al código decrecería y las funciones de control puedan enfocarse en todos los aspectos de conciliación de módulos, adherencia funcional, etc.

  - Se sugiere generar una guía resumida, a partir de la propuesta, en la que se identifique a las personas del equipo de desarrollo hacia los roles, actividades y productos, con la finalidad de exponer explícitamente la responsabilidad de cada actor dentro del proceso de desarrollo. De esta forma se generará un equilibrio en las responsabilidades de cada rol y la estructura tomará mayor dinamismo al momento de dar seguimiento a uno o varios proyectos, evitando que los tiempos o plazos establecidos para la entrega de productos y soluciones no se vean afectados debido a la interrupción o sobre carga de funciones para un solo actor que podría tener que generar gran cantidad de documentos de varios proyectos a la vez, sin ser esto parte de sus funciones. (En lo posible generar toda la documentación sugerida por esta guía de procesos para que la adaptación de la metodología RUP sea total en la realidad de la empresa y no se obvien escenarios de riesgo)

BIBLIOGRAFIA

- Rational Unified Process. IBM Company V 7.0. Copyright IBM Corp. 1987, 2004. All Rights Reserved. Producto de software
- Ahmad Shuja, Jochen Krebs (2007): *RUP Reference and Certification Guide*
- Ivar Jacobson, Grady Booch, and James Rumbaugh (1999): "*The Unified Software Development Process*"
- Per Kroll, Philippe Kruchten (2003): *Rational Unified Process Made Easy, The: A Practitioner's Guide to the RUP*
- http://www.ibm.com/developerworks/rational/library/05/0830_VanEpps/index.html
- http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf\
- http://www.dsi.uclm.es/asignaturas/42551/trabajosAnteriores/Trabajo-Guia%20RUP.pdf
- http://www.scribd.com/doc/297224/RUP.

BIOGRAFIA

Xavier Ricardo Salazar Quintana

Nacido el 18 de abril de 1983 en la ciudad de Quito, Ecuador.
Realiza sus estudios en la primaria Borja 2 Maristas, sus estudios secundarios en el Colegio San Gabriel y los superiores en la Facultad de Ingeniería, Escuela de Sistemas, de la Pontificia Universidad Católica del Ecuador.

Eduardo Andrés Velalcázar Armas

Nacido el 25 de septiembre de 1984 en la ciudad de Quito, Ecuador.
Realiza sus estudios en la primaria Borja 3, sus estudios secundarios en el Colegio Pensionado Universitario y los superiores en la Facultad de Ingeniería, Escuela de Sistemas, de la Pontificia Universidad Católica del Ecuador.